VŠB – Technická univerzita Ostrava
Fakulta strojní
Katedra automatizační techniky a řízení

VYUŽITÍ MEMS SYSTÉMŮ PRO MĚŘENÍ PROVOZNÍCH DAT

USING MEMS SYSTEMS FOR MEASURING OF OPERATING DATA

Vedoucí bakalářské práce: Ing. Jaromír Škuta, Ph.D.
Autor: Daniel Jiříček

Ostrava, 2016

VŠB - Technická univerzita Ostrava
Fakulta strojní
Katedra automatizační techniky a řízení

Zadání bakalářské práce

Student: **Daniel Jiříček**

Studijní program: B2341 Strojírenství

Studijní obor: 3902R001 Aplikovaná informatika a řízení

Téma: Využití MEMS systémů pro měření provozních dat
Using MEMS Systems for Measuring of Operating Date

Jazyk vypracování: čeština

Zásady pro vypracování:

1. Seznamte se s řídicím modulem na bázi jednočipového počítače PIC16F873A, popište možnosti jeho programování a komunikace s okolím.
2. Seznamte se s vybranými MEMS systémy (akcelerometr, gyroskop) dostupnými na katedře a jejich konfigurací.
3. Navrhněte a realizujte měření s MEMS systémy a vytvořte jednoduchý interface pro konfiguraci a sběr dat.
4. Zhodnoťte dosažené výsledky a navrhněte směr dalšího řešení.

Seznam doporučené odborné literatury:

Diplomové práce realizované na katedře 352 v letech 2012 – 2015.
HRBÁČEK, J. 2001. *Programování mikrokontrolérů PIC 16CXX.* Praha. Nakladatelství BEN - technická literatura. 112s. ISBN 80-86056-16-3.
HRBÁČEK, J. 2002. *Komunikace mikrokontroléru s okolím 1.* Praha. Nakladatelství BEN - technická literatura. 160s. ISBN 80-86056-42-2.
VLACH, J. 1997. *Počítačová rozhraní, přenos dat a řídicí systémy.* Praha, BEN-technická literatura, 1997, ISBN 80-85940-17-4.
WHITT, M. D. 2003. *Successful Instrumentation and Control Systems Design.* New York (USA): ISA, 2003. 360 p. ISBN 1-55617-844-1.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Jaromír Škuta, Ph.D.**

Datum zadání: 11.12.2015

Datum odevzdání: 16.05.2016

doc. Ing. Renata Wagnerová, Ph.D.
vedoucí katedry

doc. Ing. Ivo Hlavatý, Ph.D.
děkan fakulty

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA Fakulta strojní OSTRAVA

Místopřísežné prohlášení

Prohlašuji, že jsem bakalářskou práci „Využití MEMS systémů pro měření provozních dat“ včetně obrázků bez odkazů vypracoval samostatně pod vedením Ing. Jaromíra Škuty, Ph.D. a uvedl jsem všechny použité podklady a literaturu.

V Ostravě dne 11. května 2016

plné jméno a podpis autora

Prohlašuji, že

- jsem byl seznámen s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., autorský zákon, zejména § 35 – užití díla v rámci občanských a náboženských obřadů, v rámci školních představení a užití díla školního a § 60 – školní dílo.

- beru na vědomí, že Vysoká škola báňská – Technická univerzita Ostrava (dále jen „VŠB-TUO“) má právo nevýdělečně ke své vnitřní potřebě bakalářskou práci užít (§ 35 odst. 3).

- souhlasím s tím, že bakalářská práce bude v elektronické podobě uložena v Ústřední knihovně VŠB – TUO k nahlédnutí a jeden výtisk bude uložen u vedoucího bakalářské práce. Souhlasím s tím, že údaje o kvalifikační práci budou zveřejněny v informačním systému VŠB-TUO.

- bylo sjednáno, že s VŠB-TUO, v případě zájmu z její strany, uzavřu licenční smlouvu s oprávněním užít dílo v rozsahu § 12 odst. 4 autorského zákona.

- bylo sjednáno, že užít své dílo – bakalářská práce nebo poskytnout licenci k její využití mohu jen se souhlasem VŠB-TUO, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly VŠB-TUO na vytvoření díla vynaloženy (až do jejich skutečné výše).

- beru na vědomí, že odevzdáním své práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, bez ohledu na výsledek její obhajoby.

V Ostravě: 11. května 2016

.....

podpis

Jméno a příjmení autora práce:

Adresa trvalého pobytu autora práce

ANOTACE BAKALÁŘSKÉ PRÁCE

JIŘÍČEK, D. *Využití MEMS systémů pro měření provozních dat:* Bakalářská práce. Ostrava: VŠB – Technická univerzita Ostrava, Fakulta strojní, Katedra automatizační techniky a řízení, 2016, Vedoucí práce: Škuta, J.

Cílem bakalářské práce „Využití MEMS systémů pro měření provozních dat“ je vytvoření měřicího systému pro měření a konfiguraci jednotlivých měřicích modulů, pokud je možná, komunikujících po SPI. U gyroskopu je cílem měřit teplotu, měřit rychlost otáčení v jednotkách [°/s] a vyčíst ID s následným zobrazením či archivací vyčtených (poté převedených) dat. U akcelerometru je cílem měřit zrychlení v jednotkách [g], a to v ose x, y a z, s možností nastavení rozsahů a jiným zápisem do registrů, taktéž s možností zobrazení či archivace vyčtených dat. V závěru zhodnotím dosažené výsledky a navrhnu, jak by mohla práce pokračovat.

Klíčová slova: mikrokontrolér, programování, řízení, SPI, MEMS, Control Web

ANNOTATION OF THESIS

JIŘÍČEK, D. *Using MEMS Systems for Measuring of Operating Data:* Bachelor Thesis. Ostrava: VŠB – Technical University of Ostrava, Faculty of Mechanical Engineering, Department of Control Systems and Instrumentation, 2016, Thesis head: Škuta, J.

The goal of the submitted thesis: “Using MEMS Systems for Measuring of Operating Data” is to create measuring system for measure and configuration of each of the measuring modules, if it is possible, communicating via SPI. For the module gyroscope is the goal to measure the temperature, measure the speed of rotation in the unit [°/s] and to read ID with continued showing or archiving of the measured data. For the module accelerometer is the goal to measure the acceleration in the unit [g], and that in the axes x, y and z, with possibility to set the ranges and another one writing to the registers, also with continued showing or archiving of the measured data. Finally my purpose is to evaluate reached goals and suggest a direction for further solutions.

Keywords: microcontroller, programming, controlling, SPI, MEMS, Control Web

OBSAH

SEZNAM POUŽITÝCH ZKRATEK

ASCII	… standard pro ukládání anglické abecedy a jiných znaků (American Standard Code for Information Interchange)
COM	… označení sériového portu v operačním systému
CS	… chip select, kontrolní linka pro zvolení zařízení
DHTML/CSS	… dynamický hypertextový značkovací jazyk a kaskádové styly
DPS	… deska plošných spojů
EEPROM	… elektricky mazatelná paměť (Electrically Erasable Programmable Read-Only Memory)
I/O	… vstup / výstup (Input / Output)
I²C	… multi-masterová počítačová sériová sběrnice
LED	… dioda (light-emiting diode)
MISO	… vstup řídicího a výstup řízeného (Master In, Slave Out)
MOSI	… výstup řídicího a vstup řízeného (Master Out, Slave In)
MSSP	… modul synchronního sériového portu
PIC	… jednočipový mikrokontrolér (Peripheral Interface Controller)
RAM	… paměť s přímým přístupem (Random Access Memory)
RS232	… komunikační standard mezi PC a jinými zařízeními
SCADA	… supervizní řízení a sběr dat (Supervisory Control And Data Acquistion)
SCI	… sériový komunikační kanál (Serial Communication Interface)
SCK, SCLK, SCL	… hodinový signál
SDA	… vývod pro přenos dat
SDI	… vstup sériových dat
SDO	… výstup sériových dat,
SPI	… serial peripheral interface, slouží ke komunikaci
SS	… slave select, kontrolní linka pro zvolení zařízení
TTL	… tranzistorově-tranzistorová logika
USART	… sériový komunikační kanál
USB	… univerzální sériová sběrnice (Universal Serial Bus)
VS, VDD, V I/O	… piny pro přenos napájecího napětí
WDT	… hlídací pes (Watchdog), slouží k vypnutí při chybě

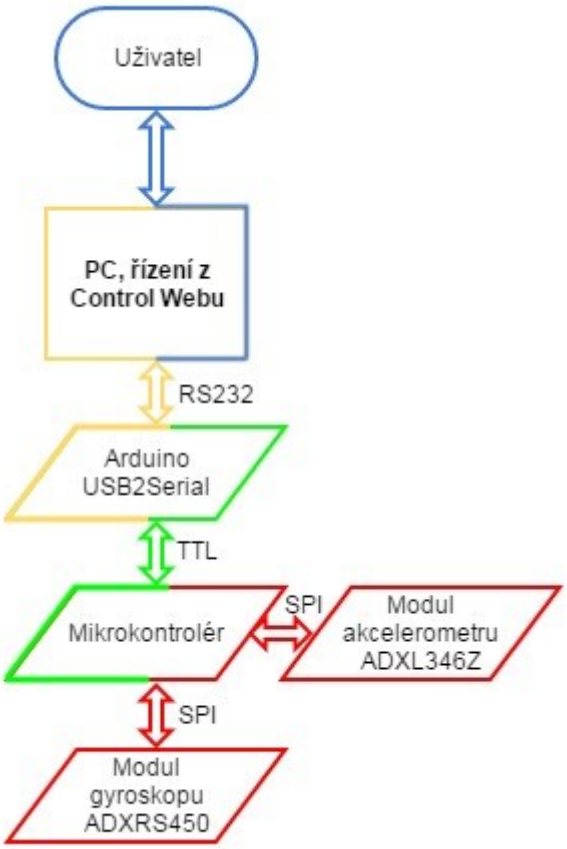
ÚVOD

Při detailním porozhlédnutí okolo sebe zjistíme, že nás dennodenně obklopují automatizované technologie, které v běžném životě běžný uživatel opravdu považuje za automatické. Dokud se jimi nevěnujeme do hloubky, nemáme potřebu vědět, jak vlastně fungují. Tyto technologie se nám staly nezbytnou součástí života. Pro představu: měření teploty v pračce nebo řízení chodu jejího motoru, jiné domácí spotřebiče (např. rychlovarná konvice, mikrovlnná trouba), řízení automobilu (např. dávkování paliva), zabezpečovací systémy, počítače, robotické linky ve firmách, měření v letadlech (např. výpočty náklonu, dávkování paliva) a spoustu dalších. To vše je realizováno za pomocí mikrokontrolérů a daných modulů, které jsou k mikrokontroléru připojeny. Také v této práci se budu mikrokontrolérem zabývat.

Cílem mé bakalářské práce je měření provozních dat na mikro součástkách zaměřených na zrychlení a natáčení a jejich grafická interpretace v softwaru Control Web. Jako první bude třeba připojit počítač k daným modulům, proto bude potřeba nastudovat parametry mikrokontroléru na desce plošných spojů. Jelikož mikrokontrolér komunikuje odlišně od počítače, bude potřeba napojit, pomocí USB rozhraní, převodník a následně na něj mikrokontrolér. Pro účely této práce je vybrán Arduino USB2Serial. Hlavní řídicí jednotkou je zvolen jednočipový mikrokontrolér PIC16F873A. Konstrukčně je tento mikrokontrolér vyřešen jako samostatná deska plošných spojů, která se dá podle svých možností programovat. K programování je využíván software MikroC. Algoritmy nestačí jen napsat. V programu MikroC bude vždy (pro nahrání na mikrokontrolér) potřeba vybudovat soubor s příponou hex, který následně v bootovacím programu PICbootPlus bude nutnost nahrát na mikrokontrolér. Pro realizaci a programování úloh je nutno na výstupy této desky připojit prvky, které chceme řídit. Bude potřeba propojit mikrokontrolér s danými moduly. Tedy za pomocí drátků a lámacích lišt, podle schéma zapojení mikrokontroléru a podle parametrů jednotlivých modulů, bude potřeba dané moduly připojit, a to přesně tam, kde to parametry vykazují. Komunikace bude tedy probíhat mezi počítačem a modulem mikrokontroléru přes Arduino USB2Serial a dále mezi mikrokontrolérem a modulem gyroskopu či modulem akcelerometru. Grafická interpretace bude, jak již bylo zmíněno, v uživatelském rozhraní ControlWeb.

VIZE

Ještě k úvodu, než se začnu zabývat danou problematikou, bych rád podotknul, že každý projekt, každá práce, zezačátku vyžadují představu o cíli. V mém případě je tato představa ve formě komunikace mezi jednotlivými zařízeními včetně koncového uživatele. Na následujícím obrázku jsem vyobrazil, pomocí jednoduché online služby Gliffy, jak by měla komunikace probíhat a jaké signály se vyskytují mezi danými objekty.



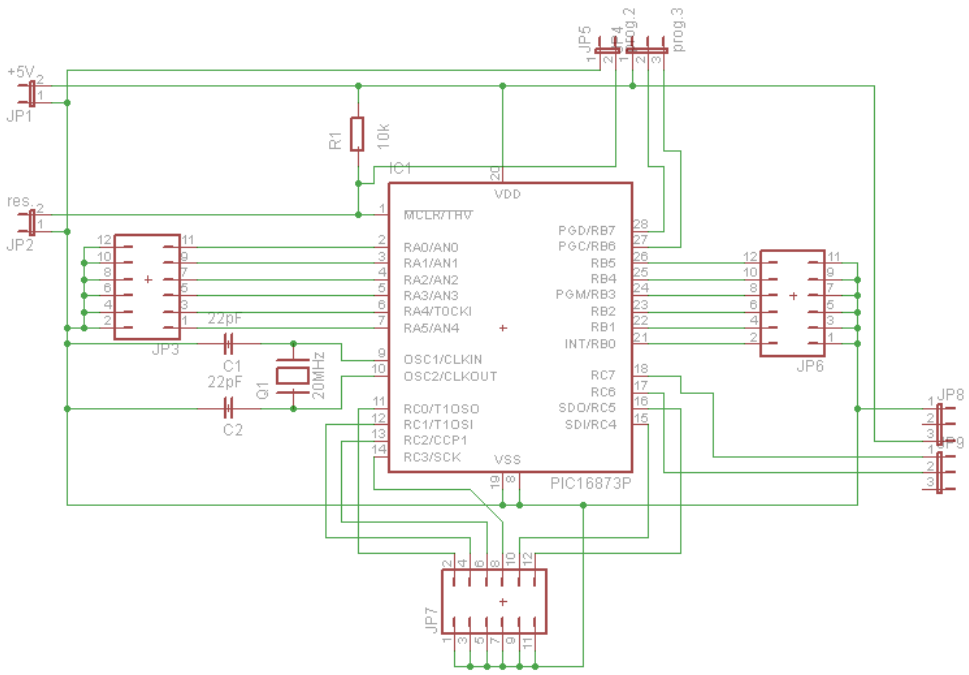
Obrázek 1: Znázornění komunikací

Na vrcholu je uživatel a jeho smysly a fyzické operace, uživatel zadá příkaz a ten se z počítače převede pomocí převodníku Arduino USB2Serial z linky RS232 na TTL logický signál, ten je dále v mikrokontroléru převeden na komunikační rozhraní SPI a zpětně vyhodnocen. Celý proces se poté od připojeného modulu opakuje zpět. Modul na požadavek od mikrokontroléru odpovídá a posílá informace zpět po SPI. V mikrokontroléru jsou informace následně převedeny zpět na TTL logický signál, který přes převodník Arduino putuje po lince RS232/USB do počítače a tam se zobrazuje uživateli.

1 MIKROKONTROLÉR

Mikrokontroléry PIC jsou programovatelné polovodičové součástky - jednočipové mikropočítače (někdy také nepříliš správně nazývané mikrořadiče nebo mikroprocesory) vyráběné firmou Microchip Technology sídlící v USA. Jsou založeny na paměti pro data a pro program. Programová paměť a datová paměť nemají stejně dlouhé datové slovo a jsou navzájem oddělené. Díky své univerzálnosti a jednoduchosti, malé velikosti a vysoké výkonnosti, nízké ceně a spotřebě mají uplatnění téměř všude. [Peroutka 1998, Hrbáček 2004]

Na následujícím obrázku je rozložení desky plošných spojů, které si můžeme souhrnně kategorizovat jako bránu A (RA0 až RA5) s piny nastavitelnými na vstup nebo výstup, bránu B (RB0 až RB7) taktéž s nastavitelnými piny na vstup či výstup a bránu C (RC0 až RC7).



Obrázek 2: Schéma DPS s PIC16F873A [ŠKUTA, J. a MAREK, H. 2006]

Program Eagle umožňuje tvorbu a zobrazení schémat nejen zapojení desky plošných spojů, ale i konkrétní propojení jednotlivých pinů, které bude potřeba pro další vývoj práce, pro připojení gyroskopu a akcelerometru k mikrokontroléru PIC16F873A.

Tabulka č. 1: PIC16F873A [ŠKUTA, J. a MAREK, H. 2006]

Typ	Paměť			I/O Pins	A/D převodník kanálů	počet časo-vačů	Periferie			Max. frek. [MHz]	počet vývodů
	Program Words	EEPROM Bytes	RAM Bytes				I2C	SPI	USART		
PIC16F873A	4096x14	128	192	22	5 (10-bit)	3+WDT	I2C	SPI	USART	20	28

Programovou paměť tvoří vnitřní paměť typu EEPROM. Datová paměť je tvořena pamětí RAM a blokem datové paměti EEPROM.

EEPROM

Datová paměť o velikosti 128 nebo 256 bytů (pozn. byte, zastarale oktet, je osmiciferné binární číslo o osmi bitech (binary digit= dvojková číslice) nabývajících hodnoty 0 nebo 1 (dvojková= binární soustava)). EEPROM je elektricky mazatelná paměť, kterou je nutné před novým zápisem nejdříve smazat (přivedením kladného napětí na adresové vodiče). Kvůli nižšímu počtu zápisů se dnes využívá častěji paměť Flash. [PAWLENKA, T 2015, Peroutka 1998]

RAM

Paměť dat je organizována do bloků o velikosti 128 bytů (do tzv. banky). Procesor obsahuje vždy 4 banky. Spodní část každé banky je použita pro speciální funkční registry, které nastavují a řídí činnost procesoru a periférií. Zbytek banky je použit jako uživatelská paměť RAM. Velikost uživatelské paměti RAM je buď 192, nebo 368 bytů.

U procesorů 876/877 je posledních 16 bytů každé banky společný všem bankám. To znamená, že můžeme k těmto 16 bytům přistupovat vždy a nezáleží na tom, ve které bance se právě nacházíme. Procesory typu 873/874 mají uživatelskou část banky 2 mapovanou na banku 0 a uživatelskou část banky 3 mapovanou na banku 1.

I/O Pins

Počet pinů nastavitelných na vstup nebo výstup je 22.

A/D převodník kanálů

10-i bitový převodník s 5-i kanálovým multiplexovaným (kombinovaným v jeden) vstupem. Doba převodu se pohybuje v řádu desítek až stovek mikro sekund na kanál.

Časovače

Počet časovačů je 3+ WDT… časovače jsou za sebou označovány jako TMR0, TMR1 a TMR2. WDT je pak watchdog (z angl. „hlídací pes“), který má funkci resetu při zaseknutí systému.

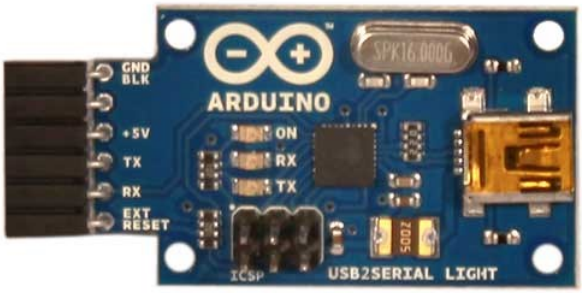
Periferie

USART/SCI je sériový komunikační kanál s možností detekce devíti bitové adresy. I²C (anglicky Inter-Integrated Circuit) je multi-masterová počítačová sériová sběrnice vyvinutá firmou Philips, obsahuje všechny funkce potřebné pro řídicí (master) i řízené (slave) režimy. SPI je komunikační sériové periferní rozhraní.

1.1 Implementované komunikační moduly

Arduino Serial Light Adapter

Pro komunikaci po sériové lince RS 232 mezi mikrokontrolérem a počítačem je potřeba mít převodník. Převodník Arduino USB2Serial převádí kanály Rx a Tx na USB.



Obrázek 3: Arduino Serial Light Adapter [ARDUINO]

Komunikace mezi mikrokontrolérem a počítačem přes tento převodník probíhá prostřednictvím komunikačních režimů v modulu synchronního sériového portu (MSSP).

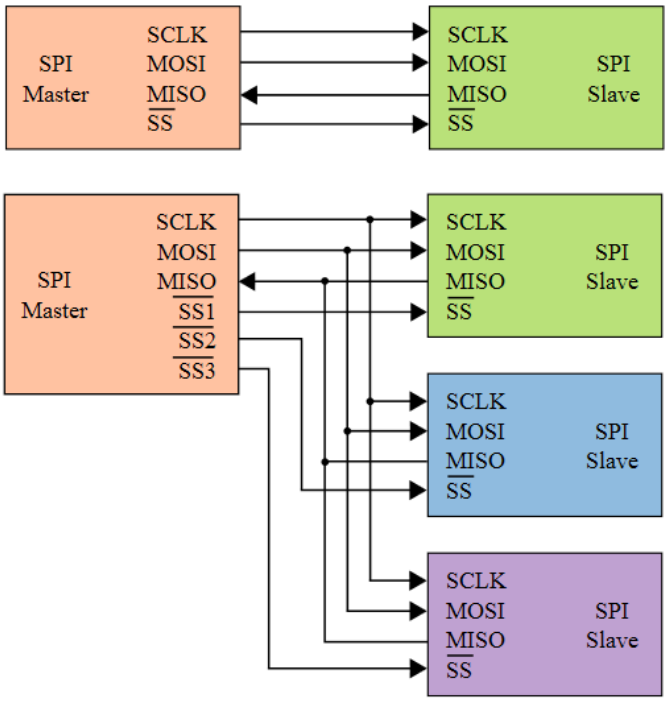
Modul synchronního sériového portu

Tento modul je v podstatě sériová linka použitelná pro komunikaci s vnějšími obvody nebo s jiným mikrokontrolérem. Funkční modul může pracovat v jednom ze dvou režimů a to buď SPI, anebo I²C, které popíši dále v této práci.

Sériové periferní rozhraní

SPI se používá pro komunikaci mezi řídicími mikrokontroléry a ostatními integrovanými obvody (EEPROM, A/D převodníky, displeje…). Komunikace je realizována pomocí společné sběrnice. Adresace se provádí pomocí zvláštních vodičů, které při logické nule na SS (respektive CS) aktivují příjem a vysílání zvoleného zařízení (CS= Chip Select). V režimu SPI je možné osmi-bitová data vysílat i přijímat současně. Ke komunikaci jsou použity 3 vývody: SDO, SDI a SCK. SDO je výstup sériových dat, SDI je vstup sériových dat a SCK je hodinový signál a 4. vývod SS může být použit pro komunikaci v řízeném režimu (SLAVE). [Peroutka 1998]

Sběrnice je rozdělena na dvě zařízení Master a Slave. Délka vysílaných dat je 8bitů (byte) nebo 16bitů (word)



Obrázek 4: SPI zapojení *[ATCEILING]*

Režim I²C

I²C (anglicky Inter-Integrated Circuit) je multi-masterová počítačová sériová sběrnice vyvinutá firmou Philips, obsahuje všechny funkce potřebné pro řídicí (master) i řízené (slave) režimy. Dva vývody SCL a SDA jsou použity pro přenos dat. Vývod SCL je použit pro hodinový signál a vývody SDA pro data. Je-li povolen režim I²C, tak jsou vývody automaticky konfigurovány. [Peroutka 1998]

Master

Master je řídicí zařízení a řídí komunikaci pomocí hodinového signálu. Pro komunikaci Master nastaví log 0 na CS zařízení, se kterým chce na sběrnici komunikovat. Pak začne generovat hodinový signál na SCLK a v té chvíli vyšlou obě zařízení svá data. Komunikace dále pokračuje a hodnota SS se nemění, anebo může být ukončena: Master přestane vysílat hodinový signál a nastaví SS do log. 1.

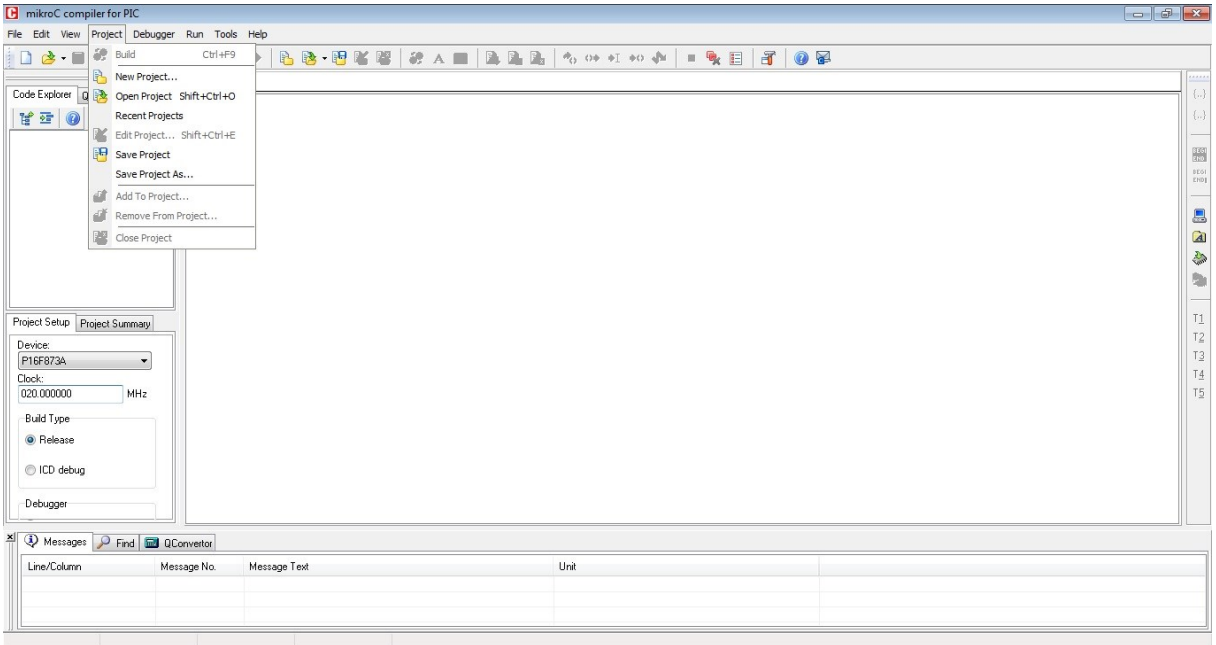
Slave

Slave je podřízené zařízení a vysílá podle hodinového signálu, pokud je aktivován pomocí CS. V řízeném režimu jsou data vysílána a přijímána, jakmile se na SCK (respektive SCLK) objeví hodinový signál, který není generován, ale pouze zjištěn a přijímán. [Peroutka 1998]

Pro účely mé práce jsem si zvolil komunikační potokol SPI.

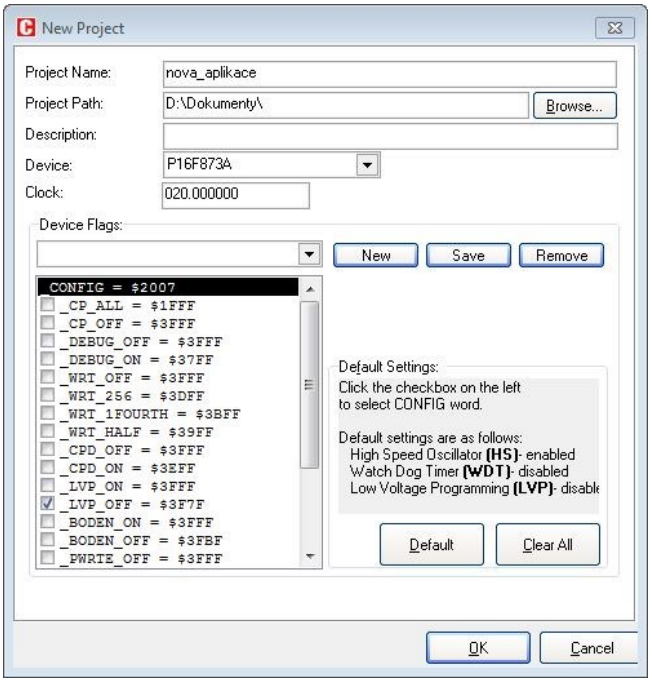
1.2 Prostředí pro programování

Dříve se hojně používalo programu Assembler se strojovými příkazy. Doba si postupně kladla nároky na programátory se sofistikovanějšími řešeními, tak se vyvíjelo i vývojové prostředí a nároky uživatelů. Programový kód píši v programu MikroC verzi 8.2.0.0.



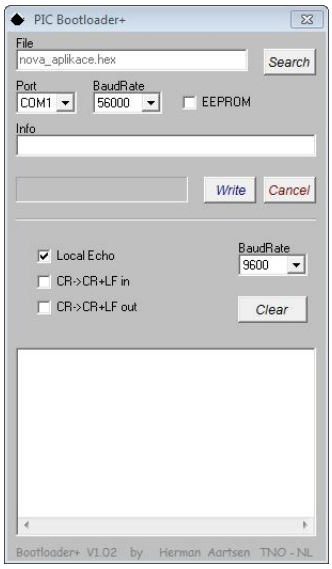
Obrázek 5: Prostředí MikroC

Na Obrázek 5 je vidět prostředí pro programování, pro založení nového projektu klikneme na New Project… a následně můžeme definovat parametry našeho mikrokontroléru, viz Obrázek 6. Nastavíme název projektu, cestu, popřípadě stručný popis, kód používaného zařízení PIC a volitelnou frekvenci krystalu, u PIC16F873A je maximum 20 MHz. Vše ostatní nastavíme tlačítkem Default.



Obrázek 6: Založení nového projektu

Po napsání programovacího kódu klikneme na Build, viz Obrázek 5, a vytvoříme tak soubor s příponou hex. Pro nahrání programového kódu (souboru hex), je potřeba tzv. Bootloader, v mém případě tedy program PICbootPlus.



Obrázek 7: PICbootPlus

Na Obrázek 7 nastavíme cestu k souboru hex, port počítače, na kterém máme připojen převodník s mikrokontrolérem a BaudRate, horní na 56 000 a spodní vpravo podle námi nadefinované USART linky v programovacím kódu. Klikneme na Write a následně na resetovací tlačítko na mikrokontroléru. Algoritmus je poté plně funkční a nahraný v zařízení.

Jednoduchá úloha na blikání LED

```
char i;
void init (void)
    {       trisb=0;                    // Nastavení brány B jako výstupní.
            Usart_Init(9600);           // Konfigurace sériové linky.              }
void main (void)
    {      init();
do  {                                   // Test výstupu.
   portb.f0=0;                          // Výstup RB0 nastaví na 0.
   delay_ms(200);                       // Časová prodleva.
   portb.f0=1;                          // Výstup RB0 nastaví na 1.
   delay_ms(200);                       // Časová prodleva. }
while (1);                              // Nekonečná smyčka.                       }
```

Jednoduchá úloha na poslání znaku A a rozsvěcení RB0

```
char i;
void init (void)
   {
   trisb=0;                             // Nastavení brány B jako výstupní.
   Usart_Init(9600);                    // Konfigurace sériové linky.
   }
void main (void)
{
init();
do
  {
  if (Usart_Data_Ready())               // Testuje, zda došel znak.
     {
      i = Usart_Read();
      Usart_Write(i);                   // Odešle znak.
      }

  if (i==65)                            // Testuje, zda přišel znak "A".
    {
     portb.f0=1;
    }
  else
    {
     portb.f0=0;
    }
  }
while (1);}
```

Tyto jednoduché úlohy poskytly informace, že je modul funkční a připraven pro další využití.

2 MEM SYSTÉMY

MEM systém je souhrnné označení pro mikro-elektro-mechanický systém, jehož prvky jsou založeny na křemíkové bázi. K tomuto označení se také pojí samotná technologie a k ní přidružená spousta produktů, které jsou založeny na elektrických nebo elektro-mechanických mikro systémech. Těmito produkty jsou například akcelerometry, gyroskopy, mikro čerpadla, mikrokontroléry, mikro cívky, mikro pohony, senzory a další.

V souvislosti s těmito produkty se hovoří o systému na čipu nebo také o inteligentním snímači, jelikož je zde přítomen jak mechanický subsystém (nutný pro transformaci fyzikální podstaty na elektrickou veličinu), tak elektronický subsystém zajišťující následné zpracování, neboli post processing (zesílení, nasycení, filtrace aj.). [ŠKUTA, J. a KULHÁNEK, J. 2012]

V dnešní době je již na denním pořádku využívání MEMS. Buď si toho jsme vědomi, nebo ne. Běžné domácí spotřebiče jako jsou pračky, rychlovarné konvice, mikrovlnky, myčky nádobí, domácí pekárny, všechny mohou uvnitř obsahovat integrované obvody MEMS.

Využití v různých oborech a odvětvích průmyslu je opravdu širokospektrální, tedy elektronika, robotika, strojnictví, chemie, medicína, a také jsou nedílnou součástí letectví, automobilového průmyslu a mnoha dalších. [PAWLENKA, T. 2015]

Na trhu je velké množství MEMS zajišťujících měření zrychlení, které se liší základními technickými parametry, jako jsou: počet os (1, 2, 3), rozsah měření (± 2g, ± 4g, ... ± 70g, ...), vzorkovací frekvence (100Hz, 200 Hz, ...), filtrací měřeného signálu, komunikační rozhraní pro přenos dat (analog, SPI, I2C), možnost triggerování vlastního měření, ... [ŠKUTA, J. a KULHÁNEK, J. 2012]

Pro účely této práce je vybrán jednoosý gyroskop ADXRS450 pro měření teploty, měření natáčení v jedné ose a akcelerometr EVAL-ADXL346Z pro snímání zrychlení ve třech osách a konfiguraci registrů.

2.1 Gyroskop

Výše této práce jsem již vysvětlil, co je MEMS. Přibližme si tedy modul ADXRS450.

Obrázek 8: ADXRS450 ***[ANALOG DEVICES]***

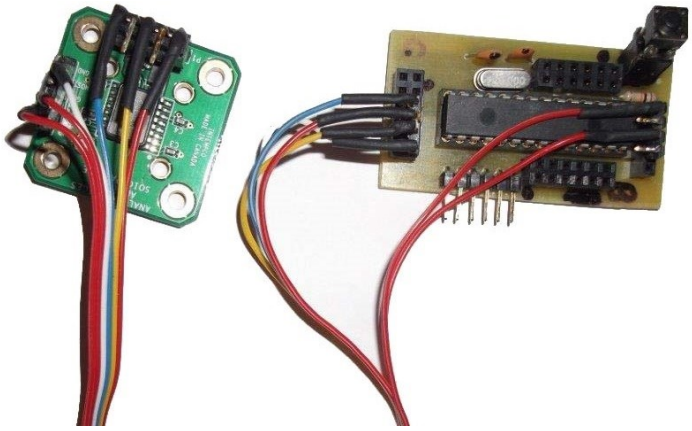
Je založen na křemíkové bázi. Obsahuje jednoosý 16-i bitový digitální výstup a pro měření natáčení musí být na desce plošných spojů. Dále má integrovaný termo senzor pro měření aktuální teploty zařízení, aby byly eliminovány případné ztráty při přenosu v závislosti na zvyšující se teplotě. Od výrobce garantovaná úhlová rychlost snímání je ± 300º/sekundu.

U jednoosých gyroskopů bývá často požadavkem možnost instalace snímače na DPS tak, aby bylo dosaženo měření ve všech třech osách. Příkladem takových pouzder je například zmíněný gyroskopy ADXRS450. [VÁGNER, M. 2015]

2.1.1 Zapojení gyroskopu

Pro komunikaci mezi gyroskopem a počítačem je třeba napojit na počítač, komunikující přes linku RS 232, převodník Arduino USB2Serial, převodník jsem napojil na desku plošných spojů s mikrokontrolérem PIC16F873A, následně jsem dle schémat z Eaglu a za pomocí pájky přiletoval dráty k lámacím lištám a propojil mikrokontrolér s gyroskopem ADXRS450.

Zapojil jsem uzemnění VSS na pin 8 (z gyroskopu GND, jako ground, země), chip select na RC2 (pin 13, z gyroskopu CS), dále jsem napojil na pin 14 hodinový signál (z gyroskopu CLK na SCK u mikrokontroléru), SDI data input na 15. pin (z gyroskopu Master In Slave Out, pro zajištění zapojení do kříže), SDO data output na 16. pin (z gyroskopu Master Oust Slave In) a napájecí napětí PDD (u gyroskopu) na pin 20 (VDD u mikrokontroléru).

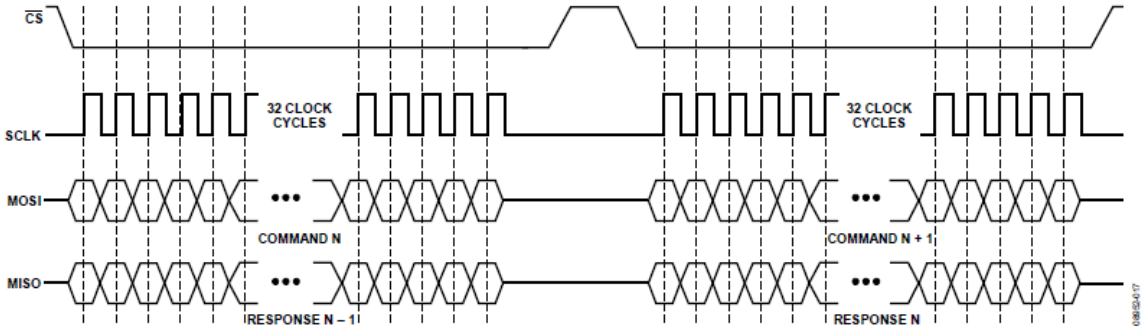


Obrázek 9: Zapojení gyroskopu k mikrokontroléru

Pro zajištění správnosti zapojení a předejití tak komplikacím s případnou následnou nekomunikací mezi zařízeními jsem ověřil za pomocí zkoušečky a osciloskopu, zda prochází lámacími lištami napětí, resp. signál.

2.1.2 Jak komunikuje s mikrokontrolérem

Použitý gyroskop komunikuje prostřednictvím 32 bitových příkazů, mikrokontrolér je schopen pomocí SPI rozhraní odesílat osmi-bitové příkazy, tzn. ve zdrojových kódech je třeba odesílat 4 příkazy po osmi bitech (byte) těsně za sebou, bez změny CS. SPI protokol gyroskopu je znázorněn na následujícím obrázku.



Obrázek 10: SPI protokol [ANALOG DEVICES, Datasheet]

2.1.3 Realizace komunikace

Sestavení příkazu je patrné z Obrázek 11. Pro účely této práce jsem používal jen příkazy určené ke čtení, tzn. první tři byty jsou 1 0 0, další tři bity jsou dané 0 0 0 a po nich následuje 9-ti bitová adresa dále 16x0 a poslední bit je paritní. Gyroskop pracuje s lichou paritou, tzn. poslední bit je doplněn tak, aby vycházel lichý počet jedniček v příkazu.

Command	Bit 31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Sensor Data	SQ1	SQ0	1	SQ2																											CHK	P
Read	1	0	0	SM2	SM1	SM0	A8	A7	A6	A5	A4	A3	A2	A1	A0																	P
Write	0	1	0	SM2	SM1	SM0	A8	A7	A6	A5	A4	A3	A2	A1	A0	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	P

Obrázek 11: Tvar příkazů pro gyroskop [ANALOG DEVICES, Datasheet]

V práci se opakují tyto tři příkazy: Žlutě zvýrazněná část příkazu je adresa bitů pro čtení.

10000000 00011000 00000000 00000000 Vyčtení ID součástky (0x5201)

10000000 00000000 00000000 00000000 Rotace kolem osy x

10000000 00000100 00000000 00000001 Teplota součástky

Command	Bit 31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Sensor Data	SQ2	SQ1	SQ0	P0	ST1	ST0	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0			PLL	Q	NVM	POR	PWR	CST	CHK	P1
Read	0	1	0	P0	1	1	1	0	SM2	SM1	SM0	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0					P1
Write	0	0	1	P0	1	1	1	0	SM2	SM1	SM0	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0					P1
R/W Error	0	0	0	P0	1	1	1	0	SM2	SM1	SM0	0	0	SPI	RE	DU									PLL	Q	NVM	POR	PWR	CST	CHK	P1

Obrázek 12: Tvar odpovědi z gyroskopu [ANALOG DEVICES, Datasheet]

Data přicházející z gyroskopu mají tvar podle Obrázek 12.

Jak znázorňuje obrázek, data přicházející z gyroskopu, jsou 32- bitová. Tvar odpovědi při čtení by měl být 0 1 0, následuje paritní bit pro první polovinu odpovědi, dále 1 1 1 0 0 0 0 a 16 bitů konkrétních dat, následuje 4x0 a paritní bit pro druhou polovinu. Tzn. při čtení ID součástky, přijde odpověď:

01011110 00001010 01000000 00100000 Odpověď na čtení ID

Kde žlutě zvýrazněných 16 bitů je podstatných a obsahuje námi hledanou hodnotu, která po převedení do hexadecimální tvaru je 5201, což přesně odpovídá údajům z datasheetu.

2.1.4 Realizace algoritmu

Řazení funkcí v MikroC má svou strukturu. Jako první přichází definice proměnných, tedy globálních proměnných. Následuje inicializace portů a komunikací, na kterých následně pracujeme. Po inicializace přicházejí jednotlivé funkce, přes které voláme interní příkazy. Jako poslední máme hlavní funkci volající dříve nadefinované jednotlivé bloky funkcí.

Jednotlivé bloky vypadají obdobně, neboť jak už jsem zmínil, je třeba odesílat 4x8bitů po SPI komunikaci. Podle datasheetu jsem zadal jednotlivé odesílané bity pro vyčtení potřebných dat zpět. Například vyčtení rotace dle následujícího algoritmu.

```
void ROT()                      //rotace okolo osy
{   for (i=0;i<2;i++)
{
portc.f2=0;                     //chip select na nulu pro komunikaci
spi_rd[0] = Spi_Read(0b10000000);
spi_rd[1] = Spi_Read(0b00000000);
spi_rd[2] = Spi_Read(0b00000000);
spi_rd[3] = Spi_Read(0b00000000);
portc.f2=1;                     //ukončení komunikace
Delay_ms(2);
}}
```

Nejdůležitějším bodem bylo vymezit žlutě ukázaných 16 bitů, to jsem realizoval následujícím algoritmem převodu s převodem na 6 rozlišitelných znaků.

```
void PREVOD(unsigned short e, unsigned short f, unsigned short g, unsigned short h)
{
mezivysledek=g;
spi_rd[0] = f<<3;                   //rotace o tři místa
spi_rd[1] = g>>5;                   //rotace o pět míst
spi_rd[2] = mezivysledek<<3;        //rotace o tři místa
spi_rd[3] = h>>5;                   //rotace o pět míst
HB = spi_rd[0] | spi_rd[1];         //logický součet
LB = spi_rd[2] | spi_rd[3];         //logický součet

  data[1]=HB/100;                   //převod dvou bytů na 6 rozlišitelných znaků
  data[2]=(HB- (data[1]*100))/10;
  data[3]=HB - (data[1]*100)-(data[2]*10);
  //LB=130;
  data[4]=LB/100;
  data[5]=(LB- (data[4]*100))/10;
  data[6]=LB - (data[4]*100)-(data[5]*10);

  data[1]=data[1]+48;               //jednotlivé převody na čísla
  data[2]=data[2]+48;
  data[3]=data[3]+48;
  data[4]=data[4]+48;
  data[5]=data[5]+48;
  data[6]=data[6]+48;}}
```

```
void main()
{   init();
    while (1)
    { if (Usart_Data_Ready())          // testuje připravenost linky
      { i = Usart_Read();              // proměnná i jako to, co přečte
      switch (i)
      {
       case 65:                        // pokud je A, obdobně B, C
         ID ();
          if (funguje==1)              // pokud funguje je jedna
           {
           PREVOD (spi_rd[0],spi_rd[1],spi_rd[2],spi_rd[3]);
           data[0]=71;                 // uvozovací znak
           for (i=0;i<7;i++)
           {  Usart_Write(data[i]);    // zápis získaných dvou bytů
              Usart_Write(13);  }      // zakončovací znak
           else
           { portb.f0=0; }
        break;                         …pokračování funkcí B a C
```

V MikroC jsem realizoval algoritmus komunikace gyroskopu s mikrokontrolérem přes linku USART a při zadání znaků gyroskop odpovídá. Vytvořený firmware (jeho část lze vidět v algoritmu výše) obsahuje v hlavní funkci (void main, řešeno pomocí case příkazů) tři funkce, které se spouštějí na základě znaků přijatých přes USART, tedy přes bootloader PICbootPLUS. Mikrokontrolér reaguje na tyto znaky:

A- Dojde k vyčtení ID připojeného gyroskopu ke sběrnici SPI. Mikrokontrolér přečte ID gyroskopu, pokud je rovno 082001 (hexa 5201), pak je připojen ADXRS450, což odpovídá hodnotě na datasheetu. B- Při přijetí tohoto znaku mikrokontrolér čte jednorázově data ze senzoru rotace a odesílá přes sériové rozhraní. C- Režim pro čtení teploty, dochází ke čtení a odesílání dat teploty.

Tvar odesílaných dat z mikrokontroléru

Data, která PIC vrací, mají následující tvar a při přijímání je třeba tyto data rozdělit. První znak je rozlišující a určuje typ dat následujících za ním. Následuje vždy šest znaků, demonstrujících číslo ve stringovém tvaru dvou bytů.

GXXXXXX Informace o připojeném gyroskopu, jeho ID

OXXXXXX Data přicházející ze senzoru rotace

TXXXXXX Data přicházející z teplotního senzoru

2.2 Akcelerometr

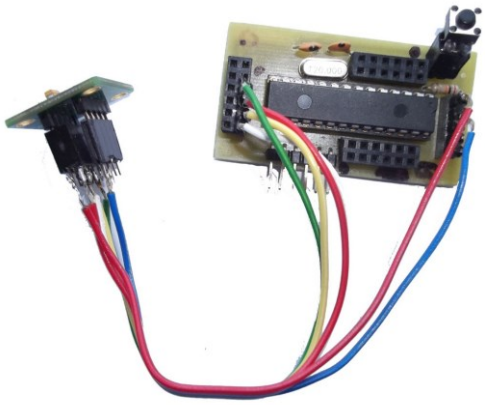
Druhým modulem je akcelerometr EVAL-ADXL346Z. Toto tříosé zařízení má funkce snímání pohybu, vibrací a rázů.

Obrázek 13: Akcelerometr EVAL-ADXL346Z [ANALOG DEVICES]

Vyniká nízkou spotřebou a nároky na procesor. Komunikace probíhá, stejně jako u gyroskopu, pomocí SPI nebo I^2C. Má pevně dané rozlišení deseti bitů. Obsahuje detekci volného pádu a také současně může detekovat čtyři až šest poloh.

2.2.1 Zapojení akcelerometru

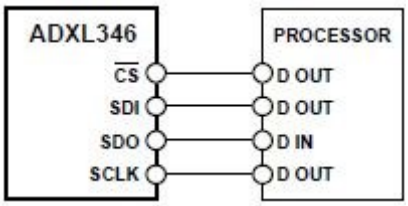
Zapojení akcelerometru taktéž obdobně jako u gyroskopu, komunikace je tedy už popsána v kapitole 2.1.1., jen doplním přesný popis. Zapojil jsem uzemnění VSS na pin 8 (z akcelerometru GND, jako ground, země), chip select na RC2 (pin 13, z akcelerometru CS), dále jsem napojil na pin 14 hodinový signál (z akcelerometru SCL na SCK u mikrokontroléru), SDI data input na 15. pin (z akcelerometru SDO, pro zajištění zapojení do kříže), SDO data output na 16. pin (z akcelerometru SDA) a napájecí napětí VIO/VDD a Vs (u akcelerometru) na pin 20 (VDD u mikrokontroléru). Také vyzkoušena správnost zapojení pomocí multimetrové zkoušečky a osciloskopu.



Obrázek 14: Zapojení akcelerometru EVAL-ADXL346Z

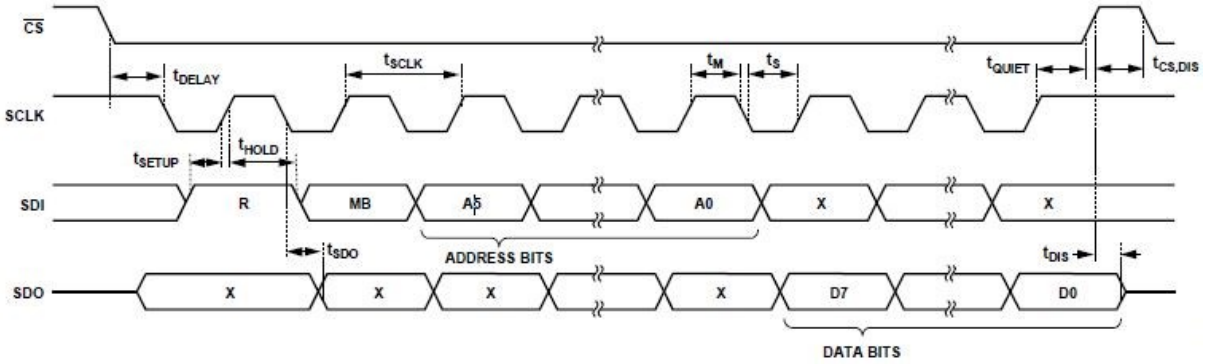
2.2.2 Jak komunikuje s mikrokontrolérem

U akcelerometru mám také na výběr mezi SPI komunikací a I^2C. Zvolil jsem komunikaci SPI. U SPI je dále rozdělení mezi tří drátovou a čtyř drátovou, v případě mého zapojení je komunikace 4-drátová, viz Obrázek 15.

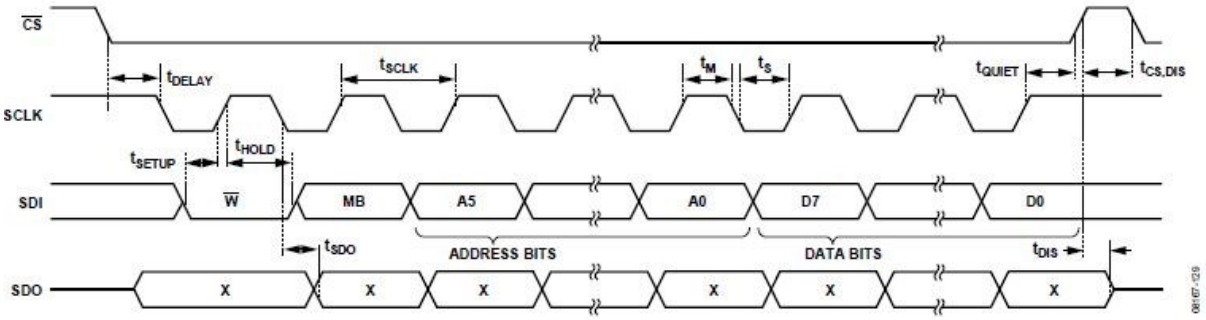


Obrázek 15: 4- drátová komunikace [ANALOG DEVICES, Datasheet]

Způsob zápisu a čtení přes SPI vypadá dle následujících dvou obrázků. Na prvním obrázku je vidět komunikace čtení dat a na druhém zápis po SPI.



Obrázek 16: 4- drátová SPI, čtení [ANALOG DEVICES, Datasheet]



Obrázek 17: 4- drátová SPI, zápis [ANALOG DEVICES, Datasheet]

Akcelerometr s mikroprocesorem komunikuje výhradně přes definované adresy registrů. Všechny příkazy jsou zobrazeny v mapě registrů na následujícím obrázku.

Address		Name	Type	Reset Value	Description
Hex	Dec				
0x00	0	DEVID	R	11100110	Device ID.
0x01 to 0x1C	1 to 28	Reserved			Reserved. Do not access.
0x1D	29	THRESH_TAP	R/W	00000000	Tap threshold.
0x1E	30	OFSX	R/W	00000000	X-axis offset.
0x1F	31	OFSY	R/W	00000000	Y-axis offset.
0x20	32	OFSZ	R/W	00000000	Z-axis offset.
0x21	33	DUR	R/W	00000000	Tap duration.
0x22	34	Latent	R/W	00000000	Tap latency.
0x23	35	Window	R/W	00000000	Tap window.
0x24	36	THRESH_ACT	R/W	00000000	Activity threshold.
0x25	37	THRESH_INACT	R/W	00000000	Inactivity threshold.
0x26	38	TIME_INACT	R/W	00000000	Inactivity time.
0x27	39	ACT_INACT_CTL	R/W	00000000	Axis enable control for activity and inactivity detection.
0x28	40	THRESH_FF	R/W	00000000	Free-fall threshold.
0x29	41	TIME_FF	R/W	00000000	Free-fall time.
0x2A	42	TAP_AXES	R/W	00000000	Axis control for single tap/double tap.
0x2B	43	ACT_TAP_STATUS	R	00000000	Source of single tap/double tap.
0x2C	44	BW_RATE	R/W	00001010	Data rate and power mode control.
0x2D	45	POWER_CTL	R/W	00000000	Power-saving features control.
0x2E	46	INT_ENABLE	R/W	00000000	Interrupt enable control.
0x2F	47	INT_MAP	R/W	00000000	Interrupt mapping control.
0x30	48	INT_SOURCE	R	00000010	Source of interrupts.
0x31	49	DATA_FORMAT	R/W	00000000	Data format control.
0x32	50	DATAX0	R	00000000	X-Axis Data 0.
0x33	51	DATAX1	R	00000000	X-Axis Data 1.
0x34	52	DATAY0	R	00000000	Y-Axis Data 0.
0x35	53	DATAY1	R	00000000	Y-Axis Data 1.
0x36	54	DATAZ0	R	00000000	Z-Axis Data 0.
0x37	55	DATAZ1	R	00000000	Z-Axis Data 1.
0x38	56	FIFO_CTL	R/W	00000000	FIFO control.
0x39	57	FIFO_STATUS	R	00000000	FIFO status.
0x3A	58	TAP_SIGN	R	00000000	Sign and source for single tap/double tap.
0x3B	59	ORIENT_CONF	R/W	00100101	Orientation configuration.
0x3C	60	Orient	R	00000000	Orientation status.

Obrázek 18: Mapa registrů pro akcelerometr [ANALOG DEVICES, Datasheet]

Na ukázku způsobu komunikace si ukážeme adresu 0x00, tedy ID akcelerometru. Po zapsání této adresy akcelerometr odpoví ve tvaru D7 až D0 bitů.

Register 0x00—DEVID (Read Only)

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	0	0	1	1	0

Obrázek 19: Odpověď na adresu 0x00 [ANALOG DEVICES, Datasheet]

Inicializační adresy

0x31 – určuje, zda má akcelerometr komunikovat s mikrokontrolérem přes 3-drátovou, nebo 4- drátovou komunikaci SPI. Jejich datový formát je na Obrázek 20.

Register 0x31—DATA_FORMAT (Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0
SELF_TEST	SPI	INT_INVERT	0	FULL_RES	Justify	Range	

Obrázek 20: Datový formát adresace 0x31 [ANALOG DEVICES, Datasheet]

kde pro nás je nejdůležitější bit D6, tedy právě nastavení 3, nebo 4- drátovou komunikaci. Kde „0“ určuje 4- drátovou komunikaci. Zbytek pole nastavíme samými nulami.

0x2D – inicializace tzv, „standby“ módu nebo měřicího módu. Pro nás nejdůležitější bit D3 (na Obrázek 21), který musíme nastavit na „1“, aby se akcelerometr přepnul do měřicího módu. Je důležité ještě zmínit, že k tomu, aby akcelerometr byl plně funkční, musí být napájen pin Vs i VDD I/O viz Obrázek 22.

Register 0x2D—POWER_CTL (Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0
0	0	Link	AUTO_SLEEP	Measure	Sleep	Wakeup	

Obrázek 21: Formát napájení [ANALOG DEVICES, Datasheet]

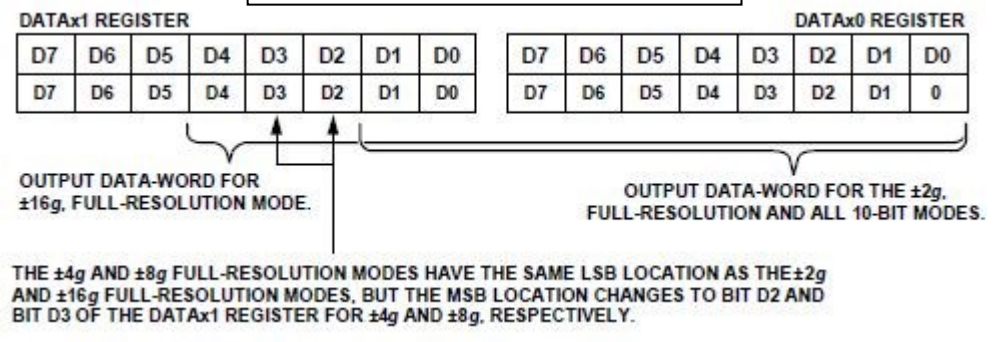
Condition	Vs	VDD I/O	Description
Power Off	Off	Off	The device is completely off, but there is a potential for a communication bus conflict.
Bus Disabled	On	Off	The device is on in standby mode, but communication is unavailable and will create a conflict on the communication bus. The duration of this state should be minimized during power-up to prevent a conflict.
Bus Enabled	Off	On	No functions are available, but the device will not create a conflict on the communication bus.
Standby or Measurement Mode	On	On	At power-up, the device is in standby mode, awaiting a command to enter measurement mode, and all sensor functions are off. After the device is instructed to enter measurement mode, all sensor functions are available.

Obrázek 22: Nastavení napájení akcelerometru a jeho popis [ANALOG DEVICES, Datasheet]

2.2.3 Realizace komunikace a algoritmu

Opět nejdůležitější částí byl převod dat. Vyřešil jsem jej podle následující části algoritmu, jelikož akcelerometr nekomunikuje pomocí 32- bitového rámce, ale pouze pomocí pevně stanoveného 10- bitového, viz Obrázek 23.

```
void prevod (unsigned hod)     // Na ascii
    {       sta= hod/100;
            hod= hod-(100*sta);
            des=hod/10;
            jed= hod-(10*des);
    data[1]=sta+48;
    data[2]=des+48;
    data[3]=jed+48;
    data[4]=sta+48;
    data[5]=des+48;
    data[6]=jed+48;
    }
```



Obrázek 23: Formát dat při zarovnání odpovědi vpravo [ANALOG DEVICES, Datasheet]

Ještě předtím je však potřeba akcelerometr správně nakonfigurovat a to zápisem do registrů. Podle mapy registrů a datasheetu viz o dvě strany dříve. Toto jsem vyřešil pomocí následující části algoritmu, při inicializaci SPI_Init_Advanced (kde nastavuji tok dat).

```
Spi_Init_Advanced(Master_OSC_div64, Data_SAMPLE_MIDDLE, CLK_Idle_HIGH, LOW_2_HIGH);
  CS=0;
  Spi_Write(READ|0x31);    // 4- vodičové zapojení
  Spi_Write(0x2D);         // měřicí mód
  CS=1;
```

Celý algoritmus poté obecně řeší obecný převod dat pro zápis a čtení do a z registrů.

```
void main (void)
{
 init();
do
  {
  if (Usart_Data_Ready())       //testuje zda došel znak
     {
     cti();
//************ WRITE ********
       if (data[0]==87)              //write
         {
         adresa=((data[1]-48)*100)+((data[2]-48)*10)+(data[3]-48);
         hodnota=((data[4]-48)*100)+((data[5]-48)*10)+(data[6]-48);

         CS=0;
         SPI_Write(adresa);
         SPI_Write(hodnota);
         CS=1;

         prevod(adresa);
         odesli();
         }
// ************ READ ********
       if (data[0]==82)            //read
         {
         adresa=((data[1]-48)*100)+((data[2]-48)*10)+(data[3]-48);
         adresa=adresa+128;

         CS=0;
         SPI_Write(adresa);
         hodnota=SPI_Read(hodnota);
         CS=1;

         prevod(hodnota);
         odesli();
        }
      }
   }while (1);
}
```

Kde odkaz na funkci odesli řeší zápis po lince USART.

3 CONTROL WEB

Systém Control Web patří do skupiny SCADA/HMI systémů. Control Web je nástroj, který slouží nejen pro vizualizaci a sběr dat, ale umožňuje i vytváření aplikací pro přímé řízení strojů v reálném čase – vytváří rozhraní člověk – stroj. [PAWLENKA, M. 2014]

Společnost Moravské přístroje a.s. založená roku 1991 se již přes 20 let soustředí na vývoj a podporu technologicky vyspělých produktů v oblasti průmyslové automatizace a elektroniky. Pyšní se dvěmi nejpoužívanějšími nástroji v těchto oblastech v ČR, a to nástrojem Control Panel a Control Web. Budu se zabývat druhým zmíněným.

3.1 O rozhraní

„Vysvětlit, co je to Control Web, a k čemu všemu se může hodit, není snadné, každé vysvětlení bude velmi dílčí a velmi neúplné. Nezabývejme se nyní tisíci konkrétními vlastnostmi a možnostmi.“ *[MORAVSKÉ PŘÍSTROJE]*

Control Web je nezávislý na hardwaru. Díky patřičným ovladačům může komunikovat s jakýmkoliv zařízením, např. PLC, I/O moduly, měřicími kartami, WWW servery a podobně. Control Web jakožto SCADA/HMI systém má možnost využívání tlustého a tenkého klienta. [MORAVSKÉ PŘÍSTROJE a.s.].

Control Web Runtime – tlustý klient:

- Sdílená data po síti přes http protokol, volání vzdálených metod
- Zálohování, synchronizace dat
- Vzdálený přístup
- Tvorba aplikací client/server nebo peer-to-peer

Přístup přes WWW prohlížeč – tenký klient

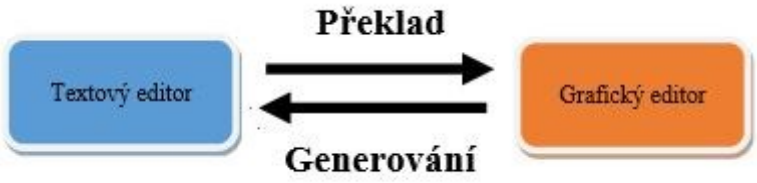
- Zabudovaný http server v systému Control Web
- Serverové aplikace pro klienty na PC i mobilních telefonech
- Umožňuje použití HTML, DHTML/CSS, Java, Active-X, atd.

Obsahuje dvě verze. Vývojovou, která je ještě dále rozdělená na vývojové prostředí v grafickém režimu a v textovém režimu, a Runtime (testovací v reálném čase). [MORAVSKÉ PŘÍSTROJE a.s.].

Systém Control Web dokáže provozovat modulární, distribuované a synchronizované aplikace. Modulární aplikace se skládá z více modulů. Systém Control Web považuje za modul každý samostatný soubor. To znamená, že nejjednodušší aplikace je jednomodulární. Taková aplikace se celá nachází v jednom zdrojovém souboru. Distribuovaná aplikace je taková, ve které jednotlivé její sdílené součásti běží na různých počítačích. Distribuovaná aplikace v systému Control Web vždy běží v modelu klient-server. Synchronizovaná aplikace je distribuovaná aplikace, ve které jsou některé její sdílené součásti udržovány v synchronním stavu (obsahují v jednom okamžiku stejná data). Nesynchronizovaná distribuovaná aplikace má přesně dáno, která součást musí být server a která klient. U synchronizované aplikace toto stanoveno není. Funkce jednotlivých distribuovaných součástí se může měnit z klienta na server nebo naopak. Vždy lze ale říci, která část je která. [PAWLENKA, M. 2014]

3.2 Vývojové prostředí

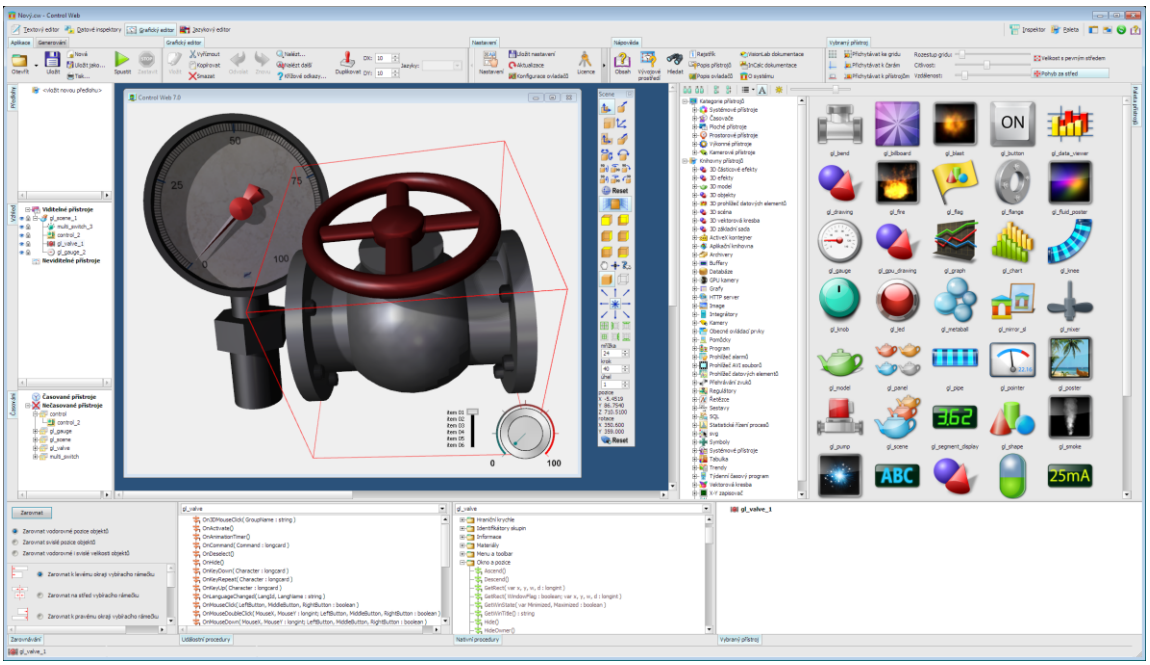
Control Web využívá dvoucestného programování, to umožňuje práci ve dvou prostředích najednou- v grafickém editoru a v textovém editoru. Změny provedené v jednom prostředí se téměř okamžitě projeví i ve druhém a naopak (tzv. překlápění).



Obrázek 24: Překlad a generování informací

Překlad zajišťuje kontrolu správnosti zdrojového kódu, napsaného v textovém editoru, a jeho následné převedení do grafického režimu. Opačný způsob, jak jsem naznačil na Obrázek 24, tedy generování, zajišťuje tvorbu zdrojového textu z grafické podoby.

Mimo vývojové a testovací okno jsou používána také okna Palety přístrojů a Inspektora přístrojů (zobrazuje vlastnosti dané komponenty a možnost tyto parametry modifikovat). Paleta přístrojů nabízí komponenty (virtuální přístroje), které myší přetáhneme do grafického editoru aplikace, v případě použití textového editoru se jakákoliv změna projeví i v grafickém.



Obrázek 25: Integrované vývojové prostředí s nástroji v konfigurovatelných záložkových plochách [MORAVSKÉ PŘÍSTROJE]

3.3 Datové inspektory

Využívají se hlavně k vytváření všech datových elementů a proměnných, se kterými pak pracují jednotlivé přístroje v grafickém editoru. Pomocí nich se nastavují ovladače pro komunikaci. Všechny datové proměnné vytvořené v datovém inspektoru jsou globální, tudíž přístupné všem použitým přístrojům a jejich procedurám. Control Web rozeznává 4 skupiny datových typů. Logické a číselné hodnoty, řetězce znaků a buffer. [PAWLENKA, M. 2014]

Rozdělení datových typů

- logické hodnoty - obsahují binární hodnotu, tedy hodnotu vyjadřující stav zapnuto/vypnuto (true / false). Datový typ - boolean.

- číselné hodnoty - čísla jsou buď desetinná (shortreal, real), nebo celá (shortint, shortcard, integer, cardinal, longint, longcard), a to buď se znaménkem (shortreal, real, shortint, integer, longint) nebo bez znaménka (shortcard, cardinal, longcard).

- řetězce znaků - skupiny písmen neomezené délky. Zapisují se do apostrofů (").

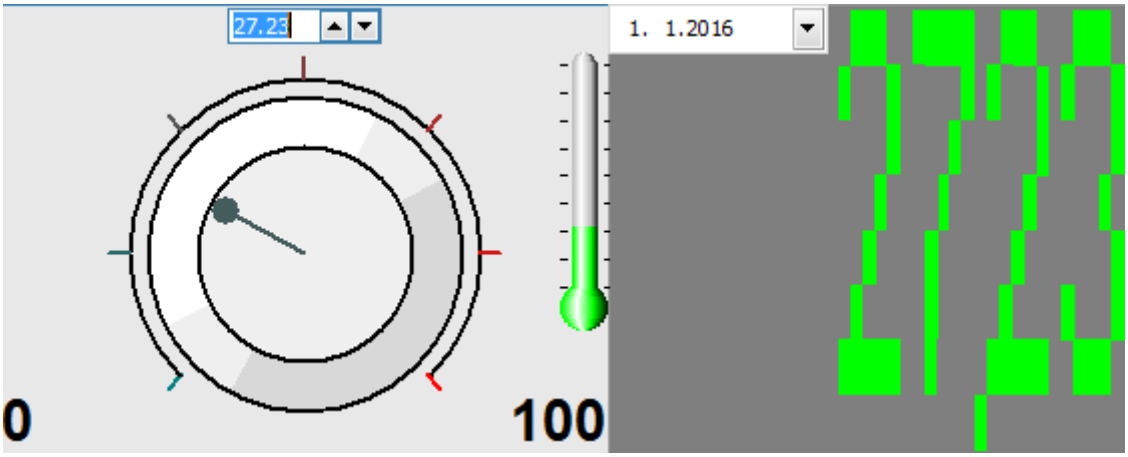
- datový typ buffer - obsahuje blok paměti, který je vnitřně strukturován. Systému Control Web definuje pouze základní parametry (například délku bufferu), zbytek vyplývá z konkrétního použití. [PAWLENKA, M. 2014]

4 ROZHRANÍ PRO KONFIGURACI ŘÍDICÍCH SYSTÉMŮ

Tento bod zadání jsem pojal ve smyslu seznámení se s interními funkcemi a provazováním prvků v rozhraní mezi sebou. Dále jsem se zabýval ovladačem, díky kterému lze komunikovat přes standardní rozhraní mezi zařízením a Control Webem. A nakonec jsem vytvořil rozhraní pro měření a sběr dat.

4.1 Jednoduchá aplikace pro seznámení

Vytvořil jsem ručně ovládanou aplikaci s kruhovým metrem, digitálním ukazatelem zapisované hodnoty, která je také znázorněna na termometru a panelem pro datum.



Obrázek 26: Control Web aplikace

Při dosažení optimální proměnné V_hodnota 25- 75 na metru (vlevo) se změní jak digitální zápis hodnoty (vpravo), tak i barva na termometru (vprostřed) na zelenou. Pod spodní hranicí limitu 25 na modrou a nad horní hranici limitu nad 75 se změní na červenou barvu.

4.2 Komunikační rozhraní

Ovladač ASCDRV5 slouží ke komunikaci mezi zařízením připojeným přes standardní sériové rozhraní počítače RS-232 (RS-485) a systémem Control Web. Díky němu lze vysílat textové řetězce (ASCII) ze systému Control Web do zařízení a zpětně přijímat textové řetězce ze zařízení, dále ovladač umožňuje řešit jednoduché asynchronní a synchronní komunikace. Mezi další vlastnosti ovladače se řadí: generování události při příjmu zprávy ze zařízení, definovatelné ukončovací znaky zpráv (terminátory) zvlášť pro příjem a vysílání, přidání předdefinovaných skupin znaků, které budou automaticky připojeny na začátek a konec vysílaného řetězce (prefix, suffix), možnost re-inicializace a změny parametrů komunikace za běhu aplikace.

Kanály ovladače

Ovladač má jednak pevně přidělenou množinu kanálů se speciálními funkcemi a dále uživatelsky definovaná pole kanálů. Tato pole umožňují přístup k jednotlivým znakům přijatého nebo vysílaného řetězce. Počáteční čísla polí kanálů ovladače se definují v sekci Settings pomocí parametrů InpBufferChannel (první číslo pole kanálů, ze kterých se dají vyčítat kódy znaků vstupního řetězce. Pole kanálů musí být číselného typu, např. real) a OutBufferChannel (první číslo pole kanálů, do kterého se dají zapisovat kódy znaků výstupního řetězce. Pole kanálů musí být číselného typu, např. real). Velikosti polí nejsou nijak omezeny. Požadovaný kanál (prvek pole) však musí být definován v souboru '*.DMF'.

Je-li zadán kanál mimo rozsah pole nebo nepadne do přijatého řetězce, ovladač vrátí hodnotu 256 (neplatný kód znaku).

Vyhrazené kanály ovladače

Kromě uživatelsky definovaných kanálů, má ovladač skupinu kanálů, které jsou pevně definovány. Tyto kanály jsou vyhrazeny pro poskytování stavových informací o ovladači a pro řízení jeho činnosti. Jsou mapovány do intervalu 1 až 99.

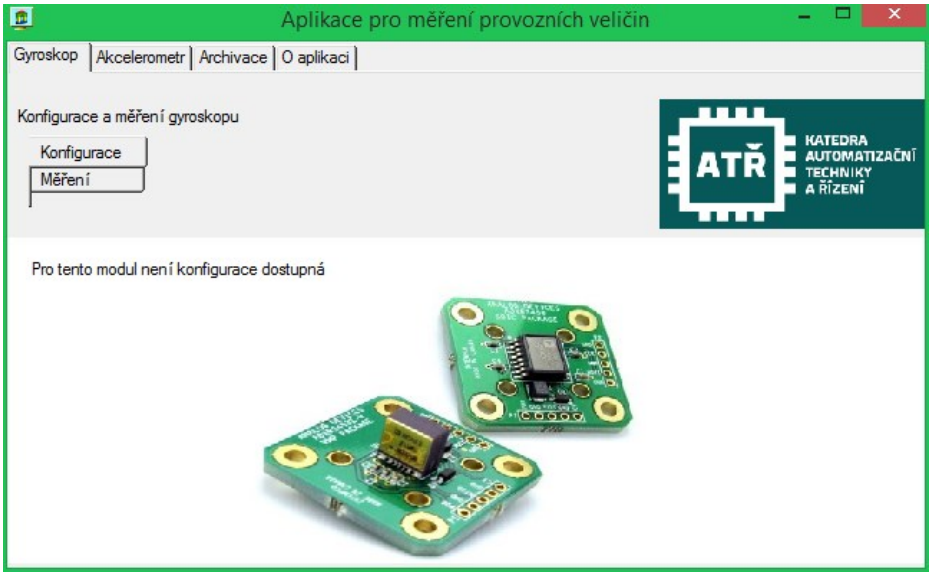
Vybrané kanály

Pro mou úlohu jsou nejdůležitější kanály č. 2, č. 20 a č. 22. První zmíněný je typu boolean output (povoluje výjimku od ovladače), druhý je typu string input (slouží k přečtení přijatého řetězce- aktuální položky z fronty) a poslední je typu string output, který je na vyslání řetězce v asynchronním módu (bez čekání na odpověď).

4.3 Měřicí rozhraní (aplikace)

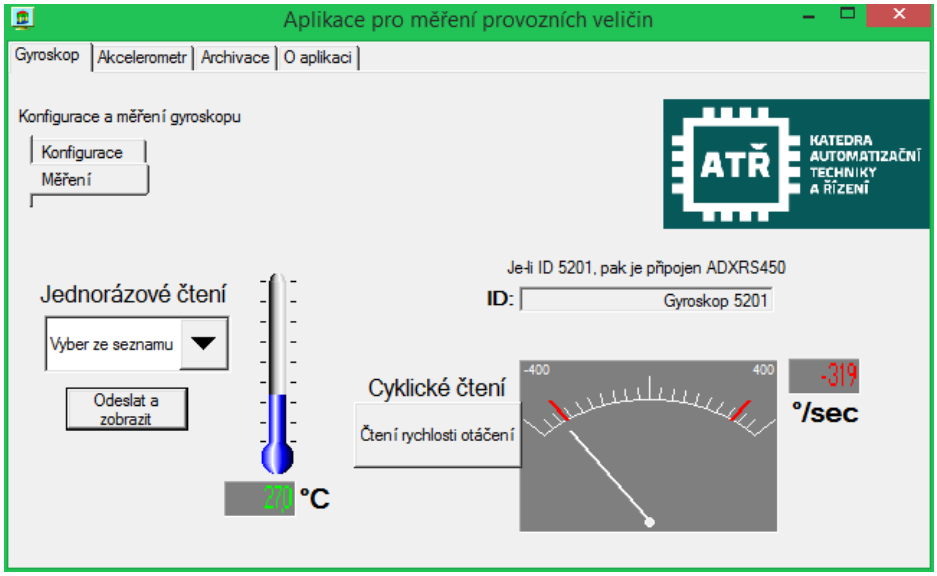
V této části práce jsem se zabýval možnostmi tvorby a ovládání různých prvků v prostředí Control Webu a také jeho ovladači, díky kterým lze komunikovat přes standardní rozhraní mezi zařízením a Control Webem. Následně jsem na základně dosažených znalostí vytvořil plně funkční uživatelskou aplikaci.

Vytvořil jsem aplikaci, ve které jsem realizoval komunikaci po SPI a USART, od uživatelského zadání příkazu, přes Arduino a mikrokontrolér až k samotným modulům a zpět.



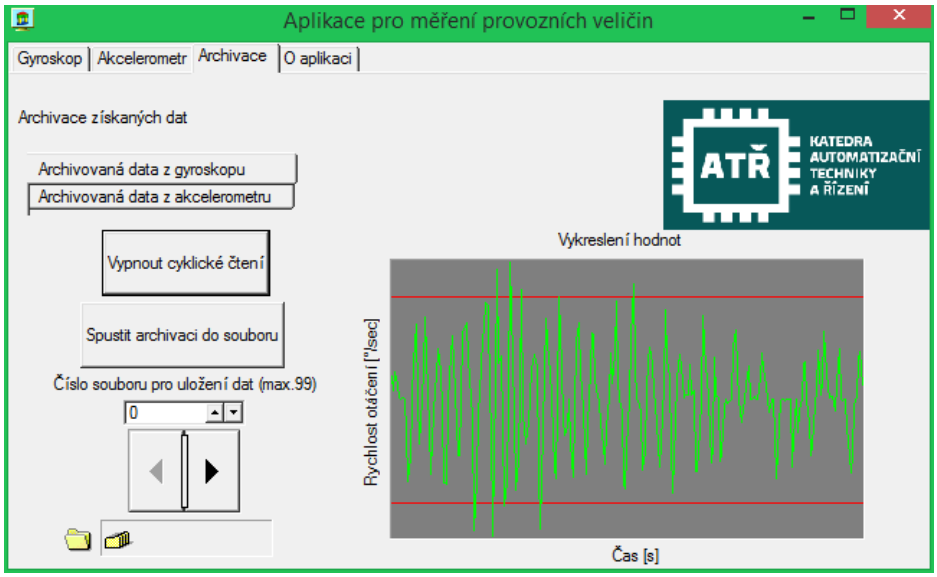
Obrázek 27: Záložka konfigurace u gyroskopu

Konfigurace pro modul gyroskopu není dostupná. Přepínání mezi záložkami jsem vyřešil pomocí objektů typu switch navázané na jednotlivé subpanely.



Obrázek 28: Aplikace v Control Webu

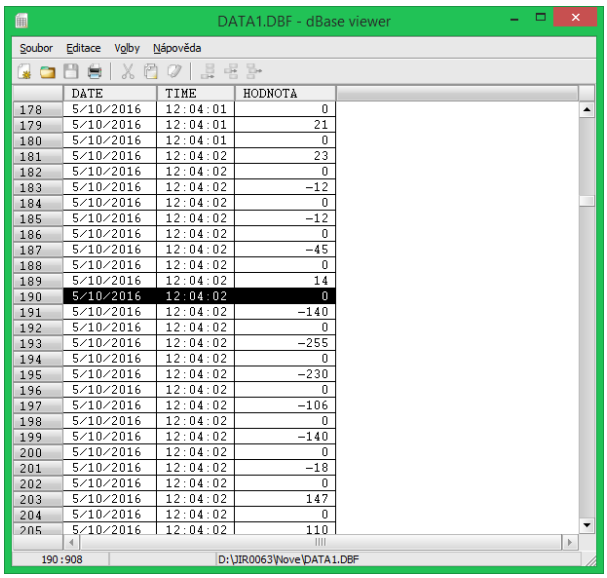
Jak už jde v aplikaci vidět, pokud chceme číst cyklicky, klikneme na tlačítko Čtení rychlosti otáčení pod titulkem Cyklické čtení. Pro jednorázové čtení je uživatel naveden na rolovací lištu, kde po vybrání ID, vyčteme ID modulu zapsané jako Gyroskop X (kde X je hodnota převedená pomocí algoritmu v Control Webu). Po výběru Rotace, jednorázově vyčteme rychlost otočení modulu ve stupních za vteřinu. Při vybrání Teplota, vyčteme jednorázovou hodnotu teploty ve stupních Celsia. To vše musíme vždy potvrdit tlačítkem Odeslat a zobrazit. Hodnoty se zobrazí na příslušných zobrazovačích.



Obrázek 29: Panel Archivace dat

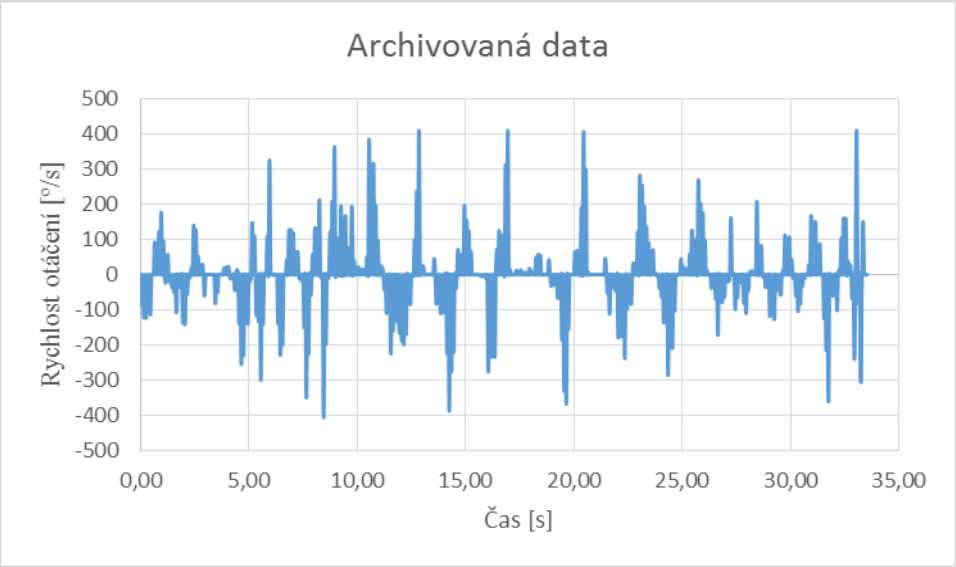
Na Obrázek 29 je panel Archivace. Vykreslení je pouze informativního charakteru s mezemi od -300 do 300 stupňů za sekundu, což je od výrobců garantovaná hodnota senzoru.

Dále při stisknutí tlačítka Zapnout archivaci, zapneme archivaci do námi zadaného souboru definovaného číslem (na Obrázek 29 číslo 0) od nuly do 99. Soubor se uloží ve stejném adresáři jako aplikace s názvem DATAXX, kde XX představuje uživatelem zvolené číslo souboru.

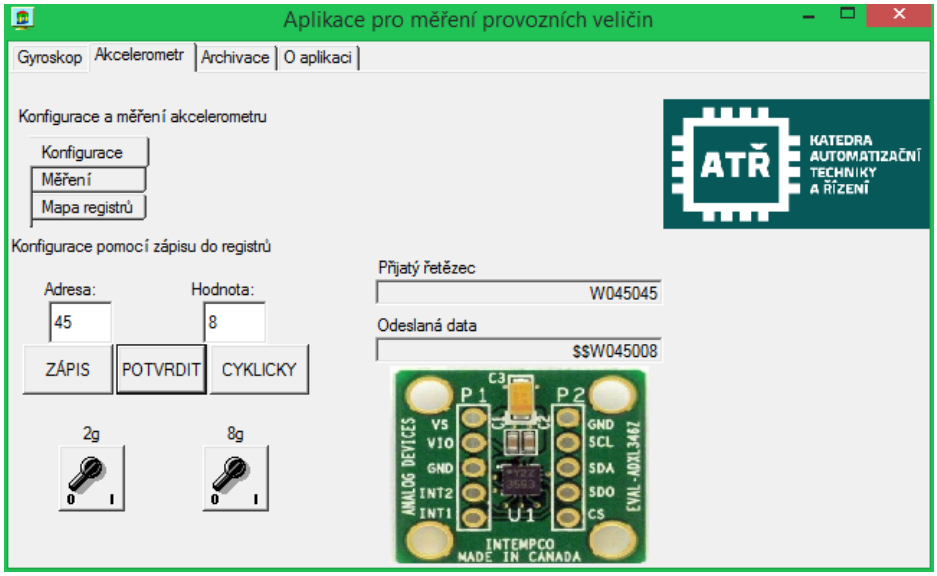


Obrázek 30: Archivovaná data

Archivovaný soubor, viz Obrázek 30, s příponou DBF zaznamenává datum, čas a hodnoty. Tento soubor lze převést do formátu excel a dále s ním pracovat jako s naměřenými daty, viz Obrázek 31.

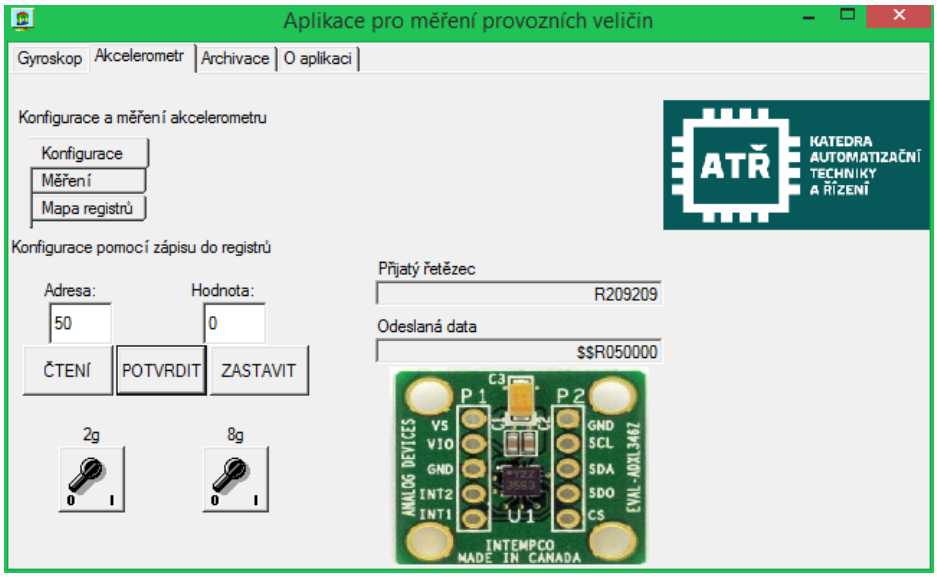


Obrázek 31: Naměřená, archivovaná data z gyroskopu



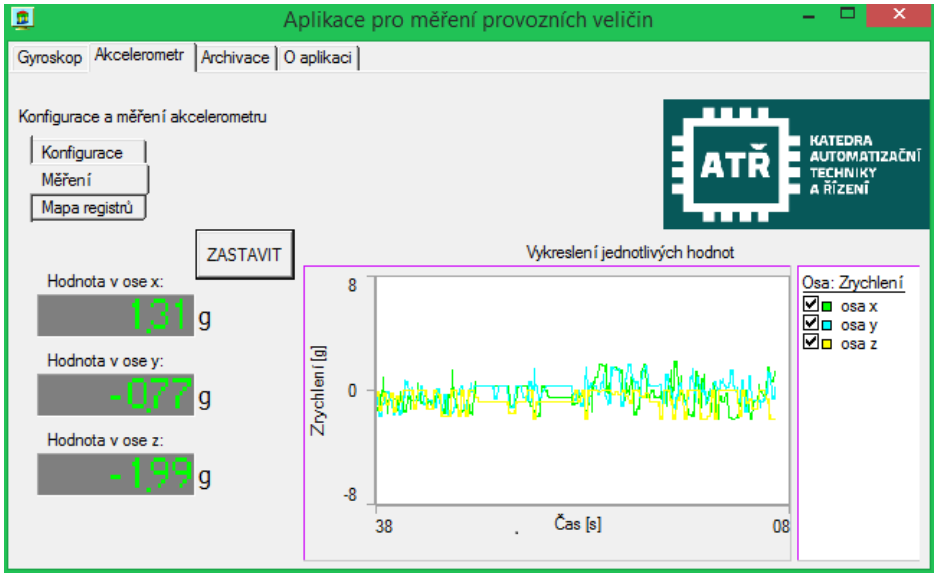
Obrázek 32: Konfigurace zápisem do registrů

Na panelu Akcelerometr na záložce Konfigurace můžeme nakonfigurovat akcelerometr. Zadáním adresy a hodnoty zapíšeme (viz Obrázek 32) nebo vyčteme (viz Obrázek 33), po stisknutí daných tlačítek, řetězec hodnot v registru. Např. adresa registru 45, hodnota 8 nám udává, dle mapy registrů, přepnutí do měřicího módu. Obrázek 32 i Obrázek 33 obsahují ilustračně zobrazený akcelerometr.



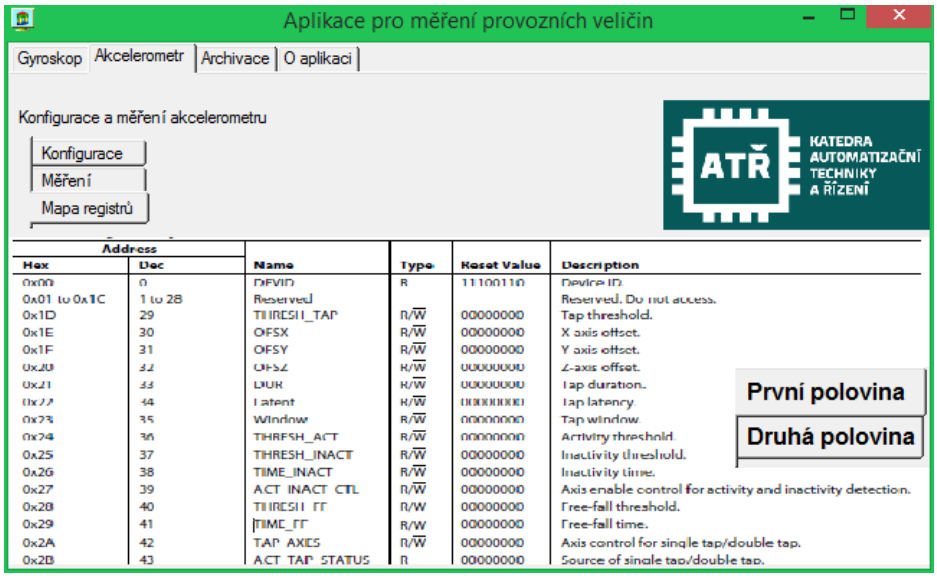
Obrázek 33: Čtení jednorázové nebo cyklické ze zadaných registrů

Na předchozím obrázku je zobrazena ta samá záložka jako při zápisu do registrů, ovšem nyní se čtením z registrů. R značí čtení a W značí zápis. Tlačítka jsou opět řešena typem switch. Na obou předchozích obrázcích jsou vlevo dole přepínače rozsahů 2g a 8g.



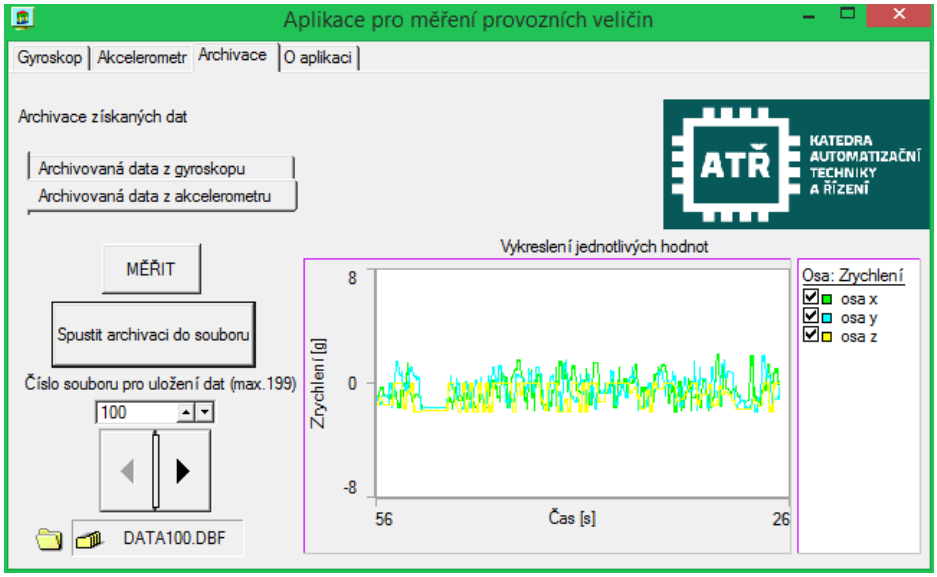
Obrázek 34: Záložka měření a zobrazení dat

Na záložce měření poté můžeme jednotlivá data zobrazovat s rozlišením jednotlivých os a vykreslením do příslušného grafu. Po pravém kliknutí na graf si dle nabídky můžeme zobrazit například nedávnou historii hodnot, různě modifikovat graf, přepínat do jiného módu apod. To vše v reálném čase, při měření a bez rizik.



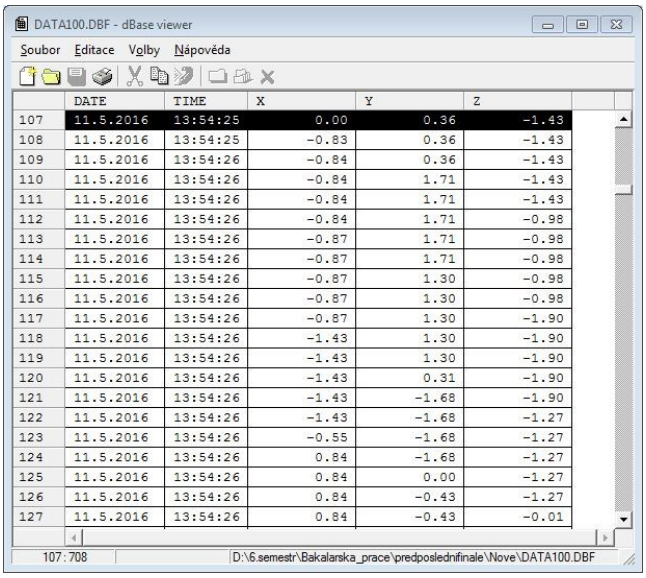
Obrázek 35: Mapa registrů pro rychlejší konfiguraci

Na Obrázek 35 je mapa registrů rozdělená na dvě poloviny, kdy si uživatel může jednoduše mezi polovinami přepínat. Tuto záložku jsem do aplikace umístil z důvodu přehlednosti a zrychlení možné konfigurace.



Obrázek 36: Panel archivace slouží k ukládání dat

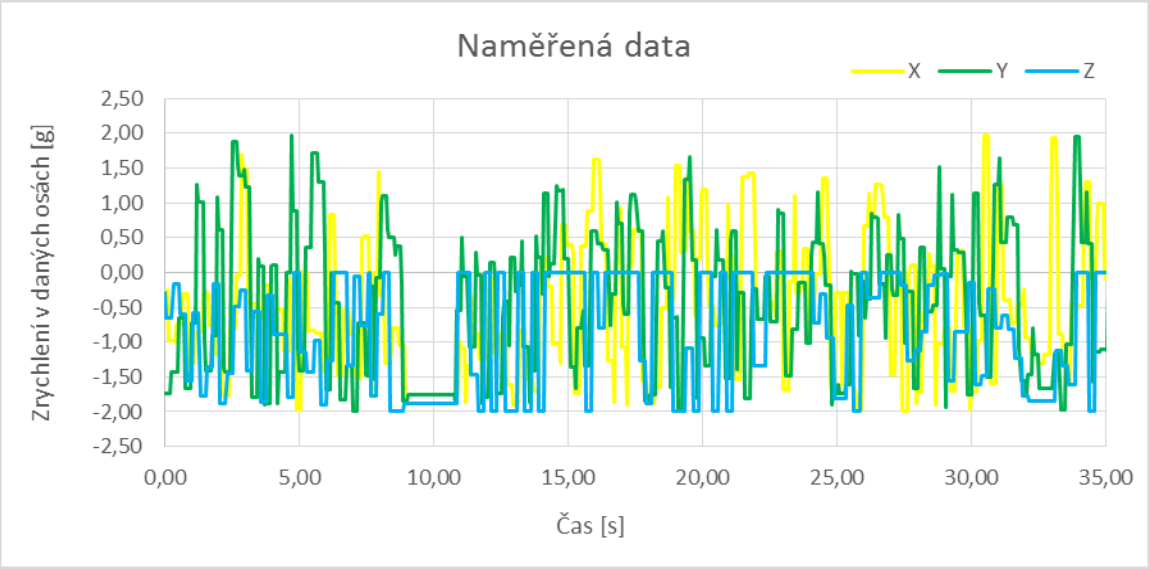
Stejně jako u gyroskopu i zde je možnost archivace dat. Tentokrát však námi zadaný soubor je možno definovat od čísla 100 do 199, aby nedošlo k přepisu dat z gyroskopu. Soubor se uloží ve stejném adresáři jako aplikace s názvem DATAXXX, kde XXX představuje námi zvolené číslo souboru.



	DATE	TIME	X	Y	Z
107	11.5.2016	13:54:25	0.00	0.36	-1.43
108	11.5.2016	13:54:25	-0.83	0.36	-1.43
109	11.5.2016	13:54:26	-0.84	0.36	-1.43
110	11.5.2016	13:54:26	-0.84	1.71	-1.43
111	11.5.2016	13:54:26	-0.84	1.71	-1.43
112	11.5.2016	13:54:26	-0.84	1.71	-0.98
113	11.5.2016	13:54:26	-0.87	1.71	-0.98
114	11.5.2016	13:54:26	-0.87	1.71	-0.98
115	11.5.2016	13:54:26	-0.87	1.30	-0.98
116	11.5.2016	13:54:26	-0.87	1.30	-0.98
117	11.5.2016	13:54:26	-0.87	1.30	-1.90
118	11.5.2016	13:54:26	-1.43	1.30	-1.90
119	11.5.2016	13:54:26	-1.43	1.30	-1.90
120	11.5.2016	13:54:26	-1.43	0.31	-1.90
121	11.5.2016	13:54:26	-1.43	-1.68	-1.90
122	11.5.2016	13:54:26	-1.43	-1.68	-1.27
123	11.5.2016	13:54:26	-0.55	-1.68	-1.27
124	11.5.2016	13:54:26	0.84	-1.68	-1.27
125	11.5.2016	13:54:26	0.84	0.00	-1.27
126	11.5.2016	13:54:26	0.84	-0.43	-1.27
127	11.5.2016	13:54:26	0.84	-0.43	-0.01

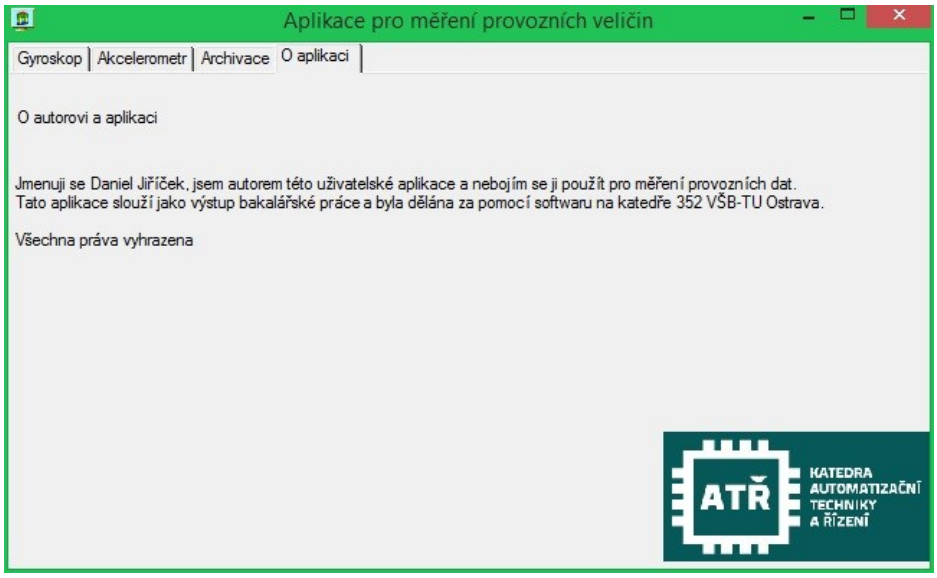
Obrázek 37: Archivovaná data z akcelerometru

Archivovaný soubor, viz Obrázek 37, s příponou DBF zaznamenává datum, čas a hodnoty ve třech osách (X, Y, Z). Tento soubor lze převést do formátu excel a dále s ním pracovat jako s naměřenými daty, viz Obrázek 38.



Obrázek 38: Vykreslení dat z excelu

Panel O aplikaci je pouze informativní v krátkosti o mě a aplikaci, viz Obrázek 39.



Obrázek 39: Panel O aplikaci

ZÁVĚR

Cílem této práce bylo vytvoření funkčního měřicího rozhraní pro komunikaci mezi jednotlivými moduly, mikrokontrolérem, převodníkem a počítačem a pro sběr dat. Řízení a monitorování dat pro koncového uživatele je prováděno přes program Control Web, kde byla vytvořena, pomocí příslušných algoritmů, aplikace pro měření a zápis dat z nebo do modulu gyroskopu nebo akcelerometru. Detailně byla rozepsaná komunikace mezi moduly a mikrokontrolérem, ve stručnosti byl popsán převodník pro komunikaci mezi mikrokontrolérem a počítačem. Byla rozebrána vnitřní struktura mikrokontroléru. Úspěšně byly také realizovány algoritmy v programovacím prostředí Mikro C. Mikrokontrolér byl naprogramován pro komunikaci s gyroskopem a následně i pro komunikaci s akcelerometrem. Řešeny byly obtížnosti v převodu jednotlivých dat a ve funkčnosti daných bloků. Programy nakonec dosáhly svého úspěchu a jsou funkční. Po řádném odzkoušení správnosti zapojení a přiletování drátků zapojených na odpovídající si piny proběhla komunikace v pořádku. Testovací úlohy vytvořené v prostředí MikroC (následně nabootovány přes PICbootPlus do mikrokontroléru) na posílání znaku a následné vracení po USART lince fungovaly. Úspěšně byly také realizovány algoritmy v programovacím prostředí Control Web, kde byly opět největší obtíže s převodem dat. V Control Webu byla vytvořena funkční aplikace pro uživatele na bázi zadávání znaků s následným zpětným vyčtením potřebných hodnot z registrů, které jsou pomocí algoritmů převáděny do srozumitelného tvaru dat pro neznalého i znalého uživatele. Pro gyroskop byla realizována záložka pro měření teploty, rotace a ID zařízení s následným zobrazením do příslušných zobrazovačů. Dále byla realizována možnost archivace těchto dat do souboru, jehož jméno si může každý uživatel definovat zadáním čísla od 0 do 99. Pro akcelerometr byla navíc možnost konfigurace, proto byla vytvořena záložka konfigurace, na které si uživatel dle mapy daných registrů (která je taktéž k dispozici v aplikaci- na záložce Mapa registrů) může akcelerometr nadefinovat jak před samotným měřením, tak i při měření. Nakonfigurování je potřeba pro prvotní nastavení měřicího módu (ten je však, z algoritmu nahraného v mikrokontroléru už defaultně nastavený) nebo pro následné vyčítání dat v jednotlivých osách atd. Na kartě Měření bylo vytvořeno real-timové měření ve třech osách současně v jednotkách g. Hodnoty v jednotlivých osách jsou na stejné kartě zobrazovány do zobrazovače dat, kde byla nadefinována vodorovná osa jako čas 30s a svislá osa jako rámec dat v jednotkách g. Příslušná legenda u grafu s rozlišením barev jednotlivých os napovídá, která hodnota patří ke zrychlení v dané ose.

Nakonec i u akcelerometru byla vytvořena záložka archivace dat, ta však není úplně nutná, protože na kartu Měření byl umístěn pokročilý zobrazovač hodnot. Jak už jsem popisoval na předchozí straně, zobrazuje všechny tři osy, pokud uživatel klikne pravým tlačítkem, tak si může zobrazit danou historii vyčítaných hodnot nebo jednoduše osy, ve kterých neměří, odkliknout. Pro případný další vývoj práce navrhuji využití gyroskopu pro sledování náklonu jiných modelů (např. sledování náklonu vrtulníku a nalézt možná řešení pro vyrovnávání), navrhuji využití akcelerometru taktéž pro účely modelu, jakým je například vrtulník. Akcelerometr by byla možnost také využít například pro měření rychlosti a rázů krokového motoru. Aplikace by mohla být v budoucnu nepochybně plně obecná se širokým záběrem modulů pro komunikaci jiných periferií přes RS232 převodník pro spojení modelu s počítačem v prostředí MATLAB/SIMULINK pomocí real-time toolboxu nebo v Control Webu.

Poděkování

Na tomto místě bych rád poděkoval Ing. Jaromíru Škutovi, Ph.D. a internímu doktorandovi Ing. Jiřímu Czebe za cenné připomínky a odborné rady, kterými přispěli k vypracování této bakalářské práce. A také děkuji vedoucím pracovníkům Katedry automatizační techniky a řízení za poskytnutí softwaru i hardwaru, potřebného k vytvoření této práce.

POUŽITÁ LITERATURA

ANALOG DEVICES. *ADXRS450* [online]. [cit. 25. 11. 2015]. Dostupné z: <http://www.analog.com>

ANALOG DEVICES. *EVAL-ADXL346Z* [online]. [cit. 25. 11. 2015]. Dostupné z: <http://www.analog.com>

ATCEILING. [online]. [cit. 17. 11. 2015]. Dostupné z: <http://4.bp.blogspot.com/-zKmnosr_nLs/Uzg0G1XYsyI/AAAAAAAAER0/6r8iOPRVy1g/s1600/SPI.png>

ARDUINO. [online]. [cit. 19. 11. 2015]. Dostupné z: < https://www.arduino.cc/en/Main/USBSerial>

BOYER, S. A. 1999. *SCADA: Supervisory Control and Data Acquisition*, 2nd Edition. New York (USA): ISA, 1999. 215 p. ISBN 1-55617-660-0.

HRBÁČEK, J. 2001. *Programování mikrokontrolérů PIC 16CXX*. Praha. Nakladatelství BEN - technická literatura. 112s. ISBN 80-86056-16-3.

HRBÁČEK, J. 2002. *Komunikace mikrokontroléru s okolím 1*. Praha. Nakladatelství BEN - technická literatura. 160s. ISBN 80-86056-42-2.

HRBÁČEK, J. 2004 *Moderní učebnice programování jednočipových mikrokontrolérů PIC – 1. díl,* BEN - technická literatura, Praha 2004, ISBN 80-7300-136-5

MORAVSKÉ PŘÍSTROJE a.s. *Control Web*. [online]. [cit. 18. 2. 2016]. Dostupné z: <http://www.mii.cz/art?id=179&cat=146&lang=405>

MORAVSKÉ PŘÍSTROJE. *Informační web firmy* [online]. [cit. 18. 2. 2016]. Dostupné z: <http://www.mii.cz>

PAWLENKA, M. 2014 *Využití PLC a PIC při realizaci modelu inteligentního rodinného domu*. Ostrava: VŠB - technická univerzita Ostrava, Fakulta strojní, Katedra automatizační techniky a řízení, 2014, Vedoucí práce: Škuta, J.

PAWLENKA, T. 2015 *Tvorba vývojového modulu pro jednočipové počítače*. Ostrava: VŠB - technická univerzita Ostrava, Fakulta strojní, Katedra automatizační techniky a řízení, 2015, Vedoucí práce: Škuta, J.

PEROUTKA, O. 1998 *Mikrokontroléry PIC16C7X* - BEN - technická literatura, Praha 1998, ISBN 80-86056-41-4

ŠOFER, P. 2008 *Využití SCADA/MMI systému pro podporu laboratorních měření.* Ostrava: VŠB - technická univerzita Ostrava, Fakulta strojní, Katedra automatizační techniky a řízení, 2008, Vedoucí práce: Škuta, J.

ŠKUTA, J. a KULHÁNEK, J. 2012 *Týmové práce na vývoji měřicího systému na bázi MEMs senzorů.* Ostrava: VŠB - technická univerzita Ostrava, Fakulta strojní, Katedra automatizační techniky a řízení, 2012

ŠKUTA, J. a MAREK, H. 2006 *Elektronické učební texty pro jednočipové procesory řady PIC* [online]. Ostrava 2006, Dostupné z: <http://352lab.vsb.cz/ServerFinalVer/Marek/pic-html/index.htm>

VÁGNER, M. 2015 *Návrh a identifikace rozšířeného modelu MEMS gyroskopu*: Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky, 2015. Vedoucí práce Beneš, P.

VLACH, J. 1997 *Počítačová rozhraní, přenos dat a řídicí systémy*. Praha, BEN-technická literatura, 1997, ISBN 80-85940-17-4.

WHITT, M. D. 2003. *Successful Instrumentation and Control Systems Design*. New York (USA): ISA, 2003. 360 p. ISBN 1-55617-844-1.